January 1996 GMU/C3I-168-IR

ADAPTIVE DECISION MAKING AND COORDINATION IN VARIABLE STRUCTURE ORGANIZATIONS

SEMI-ANNUAL TECHNICAL REPORT
Grant Number N00014-93-1-0912

for the period
1 July 1995 - 31 December 1995

Center of Excellence in Command,
Control, Communications, and Intelligence (C^3I)

George Mason University
Fairfax, Virginia 22030

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*

2. REPORT DATE
January 1996

3. REPORT TYPE AND DATES COVERED
Technical Report, 7/1/95 - 12/31/95

4. TITLE AND SUBTITLE
ADAPTIVE DECISION MAKING AND COORDINATION IN VARIABLE STRUCTURE ORGANIZATIONS

5. FUNDING NUMBERS
N00014-93-1-0912

6. AUTHOR(S)
Alexander H. Levis

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Center of Excellence in Command, Control, Communications and Intelligence
George Mason University
Fairfax, Virginia 22030

8. PERFORMING ORGANIZATION REPORT NUMBER
GMU/C3I-168-IR

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Office of Naval Research
800 North Quincy Street
Arlington, VA 22217-5660

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT
unlimited

DISTRIBUTION STATEMENT A
Approved for public release
Distribution Unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

Progress in research on coordination in distributed decision making organizations with variable structure is reported. A new approach to designing feasible organizational structures using a genetic algorithm is presented. Then, a methodology for modeling and design of distributed coordination is described. Finally, an influence diagram representation of distributed decision making is outlined.

14. SUBJECT TERMS
Decision Making; Organization theory; Colored Petri Nets; Influence Diagrams

15. NUMBER OF PAGES
68

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	**Unclassified**	**Unclassified**	**UL**

CENTER OF EXCELLENCE IN
COMMAND, CONTROL, COMMUNICATIONS AND INTELLIGENCE

GEORGE MASON UNIVERSITY
Fairfax, Virginia 22030

SEMI-ANNUAL TECHNICAL REPORT

for the period

1 July 1995 - 31 December 1995

for

ADAPTIVE DECISION MAKING AND COORDINATION IN VARIABLE STRUCTURE ORGANIZATIONS

Grant Number N00014-93-1-0912

Submitted to
Dr. W. S. Vaughan, Jr. (3 copies)
Office of Naval Research
800 North Quincy Street
Arlington, Virginia 22217-5000

Copies to:
Director, Naval Research Laboratory
Administrative Grants Office, ONR
Defense Technical Information Center

Submitted by:

Alexander H. Levis
Principal Investigator

January, 1996

GMU/C3I-168-IR

TABLE OF CONTENTS

1. PROGRAM OBJECTIVES

The objective of this research, as described in the proposal and the previous progress reports, is the investigation of several issues related to coordination in organizations. In particular, an organization is coordinated through direct and indirect means. The direct means includes the set of decision rules that the organization members use and the commands that they issue to each other. Indirect means include the dissemination of information within the organization; for example, organization members may share information or they may inform each other as to the actions they plan to take or decisions they have made. Coordination becomes a complex issue in variable structure organizations. Not only do the decision rules and the information architecture have to work for each fixed structure, but the designer has to deal with the problem, a metaproblem, of coordinating the variability. This becomes a particularly difficult problem in organizations that exhibit substantial complexity and redundancy in their information structure. The redundancy is necessary both for robustness and for flexibility and reconfigurability. In order to address these problems two main tasks were defined; they are described in the next section. In addition, some basic work in algorithms and Colored Petri Nets needs to done to develop tools and techniques for supporting the analysis and design.

2. STATEMENT OF WORK

The statement of work, as outlined in the proposal, is given below.

Task 1: Consistency and Completeness in Distributed Decision Making

Develop a methodology for analyzing and correcting the set of decision rules used by an organization with distributed decision making. The methodology is to be based on the modeling of the set of decision rules in the form of a Colored Petri Net and on the analysis of the net using S-invariant and Occurrence graphs. The ability to verify and correct the set of decision rules has direct impact on the extent of coordination needed in an actual organization and the resulting communication load.

Task 2: Variable Structures: Heuristic rules in the Lattice Algorithm Constraints

Develop a methodology for considering additional constraints in the Lattice Algorithm. Such constraints include the degrees of redundancy and complexity at the different processing nodes (to be derived from the DFS algorithm of Andreadakis), the projected response time of the organization, ands some user-specified constraints on connections between decision making units. Develop a procedure for checking the validity of such constraints and incorporate them in the Lattice Algorithm. Generalize the approach to multilevel organizational structures and to variable structures, where variable structures are obtained by folding together different fixed structures. The real focus of the task is to introduce these additional constraints as a way of containing the dimensionality problem inherent in flexibility and reducing the coordination requirements.

Design a symbolic interface for the Lattice algorithm. The interface would have the capability of interpreting natural language inputs entered by the user and will include some symbolic processing. The system will generate the interconnections matrices used as input to the Lattice algorithm. The designer would then use the various tests described in the proposal (such as DFS algorithm) to check the validity of the interconnection constraints and to make required modifications.

Task 3: Information Dissemination

Semi-annual progress reports are submitted in place of annual reports. The results of this research will appear in thesis reports and in technical papers to be presented at professional meetings and published in archival journals. In addition, oral presentations will be given periodically as arranged with ONR.

Task 4: Dynamic Task Allocation in Adaptive C2 Architectures

The objective of this task is the application of CAESAR II (Computer-Aided Evaluation of System Architectures) to several problems associated with the organizational design of flexible Command and Control Architectures for Joint Operations in modern littoral warfare. Specifically, the tools and techniques developed for the design of distributed tactical decision making organizations and for designing adaptive information structures for such organizations (as embodied in the software suite in CAESAR II). When a well defined organizational task is mapped onto the humans and machines constituting the organization, several problems of inconsistency, redundancy, and ambiguity arise that degrade organizational performance. The results of Task 1 - theory and algorithms - will be applied to this problem. Furthermore, the existence of CAESAR II with the enhancements of Task 2 makes possible the support of model-driven experimentation.

Given the nature of this task and the required close coordination with operational DOD organizations and with other organizations involved in this initiative, extensive travel is anticipated by the principal investigator and senior staff associates with this task.

Task 5: Graphical Representation of C2 Decision Making and Supporting Inference

The objective of this task is to use the rapidly developing field of influence diagrams and Bayesian networks to build graphical representations of decision making and supporting inference (intelligence) processing for command and control organizations. In particular, the emphasis is on distributed command and control organizations that are involved in rapidly evolving tactical situations and are likely to reorganize so as to remain effective. As this is our first effort in this area, our focus for this task will be on the representaiton of decision making and inference processes within command and control. The representation issues that we will adress are distributed, concurrent, asynchronous, and interconnected command and control elements; the evolution of decision making tasks throughout a typicl tactical mission; the exchange of information between elements associated with the evolution of time within a specific mission phase; and the impact of overarching the environmental and threat uncertainties that inhibit the effective paritioning of the command and control elements. This activity will lead to critical new representation ideas in influence diagrams and Bayesian networks, enabling later research that addresses partitionaing algorithms of decision making and inference tasks for the purpose of effective allocation of such tasks to command and control elements.

3. RESEARCH PLAN

The research plan describes the strategy for meeting the program objectives. Specifically the research plan is organized around a series of specific well-defined research tasks that are appropriate for theses at master's and Ph.D. levels. Individual students are assigned to each task under the supervision of the principal investigator. Additional staff from the C3I Center are included in the project whenever there is a specific need for their expertise.

4. STATUS REPORT

The focus of Task 1 is the development of a methodology for analyzing and verifying the set of decision rules used by an organization with distributed decision making. The methodology is based on the modeling of the decision rules in the form of Colored Petri Net and on the analysis of the net using S-invariant properties and Occurrence graphs. The results obtained for the two analyses, when applied to a specific form of decision rules, have been presented in the Ph. D. thesis of A. Zaidi that was submitted as the third semi-annual report. In addition, a new algorithm for designing organizations of interacting decision makers has been developed as an alternative to the Lattice Algorithm. This new algorithm is based on genetic algorithms; it is presented in section 4.1.

Task 2 has continued to be a focus of activity during this last six-month period. The recoding and revision of CAESAR II, the Computer Aided Evaluation of System Architectures suite of software algorithms and tools, has continued and the capabilities of the system are expanding. The application of CAESAR II is now Task 4 and progress in this task is reported in 4.4. A new research direction in developing models and algorithms for distributed coordination in adaptive command and control teams is described in Section 4.2.

Task 3 covers the formal efforts for documenting the results of the research as technical reports, conference papers and journal papers. The results of these efforts are presented in Section 8, Documentation.

Task 4 is a new task. This task is focused on applying CAESAR II to the Adaptive Architecturs for Command and Control (A2C2) program. Since that program is still in the early definitional stages, no major research effort was required during the last six months. This is also reflected in the project's financial reports.

Task 5 is also a new task. The results of the first stage of this effort are presented in Section 4.3.

4.1 ON GENERATING DMO ARCHITECTURES USING GENETIC ALGORITHMS (Zaidi and Levis)

4.1.1 Introduction

The methodology presented in this section is an extension of the earlier work reported in Remy and Levis (1988), Andreadakis and Levis (1988), Demaël and Levis (1994), Zaidi (1991), and Zaidi and Levis (1995) for modeling, designing and Decision Making Organizations (DMOs). As defined by Minsky (1986), distributed intelligence systems (DIS) are those systems in

which the capacity for reasoning is dispersed across its component subsystems: each function of the system is spread over a number of nodes so that each node's activity contributes a little to each of several different functions. The systems characterized as DIS carry out a number of functions, sometimes in sequence and sometimes concurrently, which makes it difficult to decompose them for their allocation to available resources. (Levis et al., 1993) The allocation of several decomposed functions to different nodes must be done in such a manner that the resulting organizational structure does not violate a number of structural and cognitive constraints.

In the work reported in aforementioned refernces, an organization is considered as a system performing a task; the system is modeled as an interconnection of organization nodes (Decision Making Units or DMUs). Each organization member is represented by a multi-stage model.(Boettcher and Levis, 1982) The formal specification of the allowable interactions between decision makers was made by Remy and Levis (1988), which led to the Lattice algorithm for generating all feasible fixed-structure architectures that meet a number of structural and user-defined constraints. Andreadakis and Levis (1988) introduced an alternative model based on the functions carried out by a resource, whether that represented a human or a machine. That model formed the basis for a different algorithm for organization design — the Data Flow Structure (DFS) algorithm. In a parallel effort, Monguillet and Levis (1993) formalized the notion of variable structure decision making organizations. Demaël and Levis (1994) extended this work and developed a methodology for modeling and generating variable structure DMOs. They presented a mathematical framework for modeling systems that adapt their structure of interactions to the input they process. Levis (1992) presented a general five stage model that subsumed all the previous ones without invalidating any of the cognitive modeling or the design algorithms. All these efforts resulted in methodologies for designing *flat* DMO architectures in which the system is viewed only from a single level of detail.(Zaidi and Levis, 1995) Although these methodologies used mathematical properties of the feasible structures to reduce the search of the solution space, when it comes to complex, and large-scale DMOs, the methodologies are confronted by the combinatorial nature of the problem.

The design methodology by Zaidi and Levis (1995) presented an approach to solve this problem by defining a DMO as families of structures with each family concerned with the behavior of the system as viewed from a different level of abstraction. The description of a DMO in a hierarchical manner (Mesarovic et al., 1970) provided a natural, structured, and modular way for formulating and solving the design problem, especially for large organizations. An organization with hundreds of lower level subsystems can be modeled with less computational effort by carefully defining the higher level subsystems of the organization

in terms of the lower level ones. The entire organization can then be modeled only in terms of the higher level subsystems. Finally, all the structures are integrated to produce a family of structures for the organization each describing the organization at different level of detail.

Although the methodology of Zaidi and Levis (1995) solves the problem by reducing the search of the feasible solutions, in doing so, it forces a designer to make certain structural decisions early in the design process, which in turn limits the degrees of freedom left for the design algorithm.

The proposed approach presents an alternative — mathematically less rigorous — for the generation of large-scale organizational structures using genetic algorithms. In the genetic algorithm procedure, an initial population of organization structures is specified, which reflects the designer's specifications. The population of structures is enhanced genetically by means of *mutation* and *crossover* operations. The newly generated structures are tested for certain structural requirements and are assigned a *fit* based on this evaluation. The *feasible* and/or *stronger* structures are retained in the population and *weaker* ones are removed. The *best* individual in the final population produced can be used as the solution to the design problem. Figure 4.1.1 presents a top-level description of the genetic algorithm (the description in the figure describes most algorithms.) (Davis, 1991) The rest of this section describes the details of the implementation of this approach for the design of DMO structures.

1. **Initiate a Population of chromosomes.**
2. **Evaluate each chromosome in the population.**
3. **Create new chromosome by mating current chromosomes; apply mutation and recombination as the parent chromosomes mate.**
4. **Delete members of the population to make room for the new chromosomes.**
5. **Evaluate the new chromosomes and insert them into the population.**
6. **If time is up, stop and return the best chromosome; if not, go to 3.**

Figure 4.1.1 Top-level Description of a Genetic Algorithm

4.1.2 Mathematical Model

The mathematical formulation of the design problem presented in Remy and Levis (1988), Demaël and Levis (1994), and Zaidi and Levis (1995) is based on Petri net theory. This paper does not describe the Petri net formalism; it has been presented in previous technical reports. The Petri net representation of the five stage decision making unit (DMU) introduced by Levis (1992) is shown in Figure 4.1.2. The labels SA, IF, TP, CI and RS are generic names for the *situation assessment*, *information fusion*, *task processing*, *command interpretation*, and *response selection* processes respectively. A DMU receives input or data x from the external environment (sensors). The incoming data are processed in the situation assessment (SA) stage to get the assessed situation z. This variable may be sent to other DMU. If the DMU receives assessed data from other DMU, these data z' are fused together with its own assessment z in the information fusion (IF) stage to get the revised assessed situation z''. The assessed situation is processed further in the task processing (TP) stage to determine the strategy to be used to select a response. The variable v contains both the assessed situation and the strategy to be used in the response selection stage. A particular DMU may receive a command v' from super-ordinate DMU. This is depicted by the use of the command interpretation (CI) stage. The output of that stage is the variable w which contains both the revised situation assessment data and the response selection strategy. Finally, the output or the response of the DMU, y, is generated by the response selection (RS) stage.

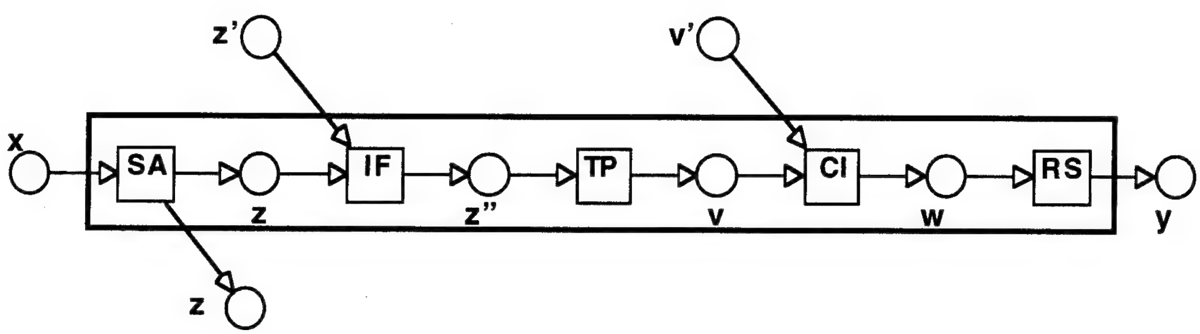

Figure 4.1.2 Five Stage Model of a DMU

As mentioned and discussed in Remy and Levis (1988), Zaidi (1991), and Zaidi and Levis (1995), only certain types of interactions make sense within the model. They are depicted in Figure 4.1.3. For the sake of clarity, only the links from the ith DMU to the jth DMU are presented. The symmetrical links from j to i are valid interactions as well. The binary variable e_i represents the *input* to a decision making node. The presence of such a link characterizes the

fact that a particular DMU may receive data from the external environment. The binary variable s_i represents the *output* of a decision making node to processes external to the organizational structure considered. The binary variable F_{ij} depicts the transmission of assessed situation from node i to node j; G_{ij} models the transmission of control from the output of a decision making node to the input of another; H_{ij} models the result or processed information sharing type of interaction between two decision making nodes; and C_{ij} represents the flow of instructions or commands from one decision making node to another.

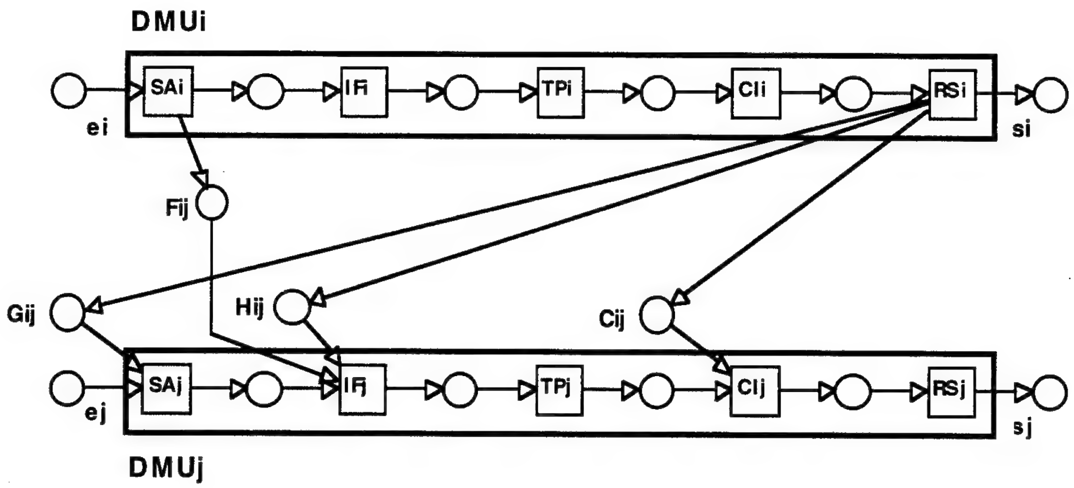

Figure 4.1.3 Allowable Interactions

The variables e_i, s_i, F_{ij}, G_{ij}, H_{ij}, C_{ij} in Figure 4.1.3 are binary variables taking values in {0, 1}, where 1 indicates the presence of the corresponding link in the organizational structure. Note that the value of the variable does not indicate the number of such links which actually exist. The variables are aggregated into two vectors **e** and **s**, and four matrices **F**, **G**, **H**, and **C**. The interaction structure of an organization consisting of n DMUs is, therefore, represented by the tuple:

$$\Sigma = \{ \mathbf{e}, \mathbf{s}, \mathbf{F}, \mathbf{G}, \mathbf{H}, \mathbf{C} \} \tag{1}$$

where

e and **s** are n x 1 arrays representing the interactions of the n-DMUs.

$$\mathbf{e} = [e_a] \quad \mathbf{s} = [s_a] \qquad a = 1,2,\ldots,n \tag{2}$$

F, **G**, **H**, and **C** are four n x n arrays representing the interactions among the DMUs of the organizational structure.

$$\mathbf{F} = [F_{ab}] \quad \mathbf{G} = [G_{ab}] \quad \mathbf{H} = [H_{ab}] \quad \mathbf{C} = [C_{ab}] \qquad b = 1, 2, ..., n \tag{3}$$

The diagonal elements of the matrices **F**, **G**, **H**, and **C** are set identically equal to zero; DMUs are not allowed to interact with themselves.

$$F_{aa} = G_{aa} = H_{aa} = C_{aa} = 0 \qquad a = 1, 2, ..., n \tag{4}$$

These relations must hold true for all solutions.

Encoding

The application of genetic algorithm to the problem of generating DMO architectures requires an encoding of the solutions in the problem domain to chromosomes—individuals in a population. The mechanism used for encoding takes the analytical model of an organization's interactional structure, $\Sigma = \{ \mathbf{e}, \mathbf{s}, \mathbf{F}, \mathbf{G}, \mathbf{H}, \mathbf{C} \}$, and converts these vectors and matrices into bit strings representing Σ. The following example illustrates the technique.

Example 1

Let the organizational structure of a DIS is given by the tuple Σ.

$$\Sigma: \quad \begin{array}{cc} \mathbf{e} = [0 \quad 1] & \mathbf{s} = [1 \quad 1], \\ \mathbf{F} = \begin{bmatrix} \# & 0 \\ 1 & \# \end{bmatrix} & \mathbf{G} = \begin{bmatrix} \# & 0 \\ 1 & \# \end{bmatrix} \\ \mathbf{H} = \begin{bmatrix} \# & 0 \\ 0 & \# \end{bmatrix} & \mathbf{C} = \begin{bmatrix} \# & 1 \\ 0 & \# \end{bmatrix} \end{array}$$

The bit string representing the chromosome, Σ, is obtained by the following encoding:

$$\Sigma \quad : \overbrace{0 \quad 1}^{e} \quad \overbrace{0 \quad 1}^{F} \quad \overbrace{0 \quad 1}^{G} \quad \overbrace{0 \quad 0}^{H} \quad \overbrace{1 \quad 0}^{C} \quad \overbrace{1 \quad 1}^{s}$$

The diagonal elements of the matrices **F**, **G**, **H**, and **C** are ignored in the bit string representation since they remain zero (4) throughout the design procedure.

Notation

An i^{th} bit in the bit string representation of an organizational structure, Σ, is accessed through the notation $\Sigma[i]$, i.e., $\Sigma[4] = 1$ in Example 1.

The length of the string representing an organizational structure Σ is denoted by $|\Sigma|$, i.e., $|\Sigma|=12$ in Example 1. The length of the bit string representing an organizational structure with n DMUs is given by:

$$|\Sigma| = 4n^2 - 2n \quad \text{where n is the number of DMUs in } \Sigma \tag{5}$$

Therefore, the index 'i' in $\Sigma[i]$ takes on the values; $1 \le i \le 4n^2 - 2n$.

4.1.3 Design Requirements

The interactional requirements for a DMO can be translated into requirements on the arrays. The designer may rule in or rule out some of the links by putting 1's and 0's at corresponding places in the arrays. This introduces the notion of *user-defined* constraints (R_u). The user-defined constraints for an organization, in terms of its constituent DMUs, are given as the tuple

$$\Sigma_i: \quad \begin{array}{cc} \mathbf{e} = [1 \quad x] & \mathbf{s} = [0 \quad x] \\ \mathbf{F} = \begin{bmatrix} \# & 1 \\ x & \# \end{bmatrix} & \mathbf{G} = \begin{bmatrix} \# & 0 \\ 0 & \# \end{bmatrix} \\ \mathbf{H} = \begin{bmatrix} \# & x \\ x & \# \end{bmatrix} & \mathbf{C} = \begin{bmatrix} \# & 0 \\ 0 & \# \end{bmatrix} \end{array}$$

The bit string representation of Σ_i is given as:

$$\Sigma_i: 1 \quad x \quad 1 \quad x \quad 0 \quad 0 \quad x \quad x \quad 0 \quad 0 \quad 0 \quad x \tag{6}$$

The x's in the arrays represent the unspecified elements or optional links. The optional links determine the degree of freedom left in the design process, and potentially yield a number of candidate solutions to the design problem, all satisfying the user-defined constraints (R_u).

Initial Population

The chromosome in (6), which represents user-defined constraints, shows the building block or *schema* (Davis, 1991; Holland, 1975) for the generation of future populations of structures. A 1 or 0 at any position means that the chromosomes in future populations must have the same value at that position for them to belong to the schema. The x's represent the genes (interactions) that can be replaced by either 1's or 0's genetically to generate new populations of solutions.

The first step in the genetic algorithm approach requires an initial population of chromosomes to start the process. In the approach presented in this paper, the bit strings representing the Universal and the Kernel Nets (Remy and Levis, 1988) are used to initialize the population.

Definition: The Universal Net associated with the constraints R_u - $\Omega(R_u)$ - is the net defined by the tuple Σ obtained by replacing all undetermined elements of $\{\mathbf{e}, \mathbf{s}, \mathbf{F}, \mathbf{G}, \mathbf{H}, \mathbf{C}\}$ by 1. Similarly the Kernel Net - $\omega(R_u)$ - is the net obtained by replacing the same undetermined elements by zero.

Definition: The bit strings representing $\Omega(R_u)$ and $\omega(R_u)$ are termed as Universal and Kernel Chromosomes respectively.

For the illustrative example in (6) the initial population is given as:

$$\begin{array}{ll} \Omega i\,(R_u): & 1\ 1\ 1\ 1\ 0\ 0\ 1\ 1\ 0\ 0\ 0\ 1 \\ \omega_i(R_u): & 1\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0 \end{array} \qquad (7)$$

4.1.4 Structural Requirements

The degrees of freedom left in the design procedure result in a very large set of candidate organizational structures. However, a number of them may correspond to patterns of interactions among DMUs that do not make physical sense. This leads to the definition of structural constraints (R_s). The structural constraints are defined with respect to the Petri net representation and the analytical model presented in (1).

Structural Constraints

(R1) The Ordinary Petri net that corresponds to Σ should be connected, i.e., there should be at least one (undirected) path between any two nodes in the net. A directed path should exist from the source place to every node of the net and from every node to the sink.

(R2) The Ordinary Petri net that corresponds to Σ should have no loops. i.e., the structure must be acyclic.

(R3) In the Ordinary Petri net that corresponds to Σ, there can be at most one link from the RS stage of a DMU i to another DMU j, i.e., for each i and j, only one element of the triplet $\{G_{ij}, H_{ij}, C_{ij}\}$ can be non-zero. The analytical expression of this constraint is given as:

$$\forall(i,j) \quad G_{ij} + H_{ij} + C_{ij} \leq 1 \qquad i \neq j;\ i, j = 1, 2, \ldots, n \qquad (8)$$

(R4) Information fusion can take place only at the IF and CI stages. Consequently, the SA stage of a DMU can either receive information from the external environment, or an output from another DMU. The translation of this constraint into mathematical terms follows:

$$\forall j \quad e_j + \sum_{i=1}^{n} Gij \leq 1 \qquad j = 1, 2, ..., n \tag{9}$$

The first part of constraint R1 eliminates any organizational form that does not represent a single structure. The second part of R1 insures that the flow of information is continuous within the organizational structure. It eliminates internal sink or source places. Constraint R2 allows acyclical organizational structures only. This restriction is imposed to avoid deadlocks and infinite circulation of messages within the organization. Constraint R3 indicates that it does not make sense to send the same output to the same DMU at several stages. It is assumed that once the output has been received by a DMU, this output is stored in its internal memory and can be accessed at later stages. Constraint R4 has to do with the nature of the IF stage. The IF stage has been introduced explicitly to fuse the situation assessments from other DMUs. It prevents a DMU from receiving more than one input at the SA stage.

Evaluation Function

The evaluation function, used in the methodology, evaluates an individual chromosome in a population on the basis of the structural requirements described in the previous section. This section describes the encoded structural constraints which check the feasibility of chromosomes against these requirements.

(R1) As pointed out in (Remy and Levis, 1988), a DMU, based upon the inputs and outputs, can have one of the four possible internal structures.

- SA alone with $y = z$
- SA, IF, TP, CI, and RS
- IF, TP, CI, and RS with $x = z'$
- CI and RS with $x = v'$ (10)

The genetic implementation of R1 checks the bit string representation of Σ to establish the internal structure of each constituent DMU. For a connected organizational architecture the internal structures of all DMUs in a Σ must fall within the four possibilities described. In addition to checking the Σ for internal structures in (9), the following, (11) and (12), checks

are also performed to ensure that the organization is also connected to the external environment through inputs and outputs (sink and source places.)

$$\sum_{j=1}^{n} \Sigma[j] \geq 1 \tag{11}$$

$$\sum_{j=1}^{n} \Sigma[4n^2 - 3n + j] \geq 1 \tag{12}$$

(R2) The manner in which allowable interactions are defined among DMUs ensures the fact that a cycle in an organizational structures must contain RS stages. An algorithm is implemented that selects each RS stage present in the organization structure and identifies recursively all stages that are input to the selected RS_i stage. In case the algorithm encounters an RS stage twice during its search, it reports the presence of a cycle in the structure, otherwise it terminates at the sink place. The algorithm keeps a record of all the RS stages that are encountered while searching loops for the RS_i stage. In the next iteration, the algorithm confines itself to only those RS stages that have not been encountered so far. The algorithm repeats itself until all such stages are searched for loops and terminates with a report of presence or absence of cycles in the structure.

(R3) The analytical expression of this constraint, applied to the bit string representation of a structure, is given as:

$$\forall i \quad \Sigma[n^2 + i] + \Sigma[2n^2 - n + i] + \Sigma[3n^2 - 2n + i] \leq 1 \qquad i = 1, 2, \ldots, (n^2 - n) \tag{13}$$

(R4) The translation of this constraint for the bit string representation of Σ follows:

$$\forall j \quad \Sigma[j] + \sum_{\substack{i=1 \\ i \neq j}}^{n} \Sigma[k_{i,j}] \leq 1 \qquad j = 1, 2, \ldots, n$$

where

$$\begin{aligned} k &= n^2 + (i-1)(n-1) + j - 1 && \text{for} \quad i < j \\ k &= n^2 + (i-1)(n-1) + j && \text{for} \quad i > j \end{aligned} \tag{14}$$

Fitness Value

Once the individuals in a population are checked by the objective function, a fitness value is assigned to each chromosome. The fitness value assigned to an individual is calculated by the following formula.

$$Fitness = \delta_f + \mathrm{NL}$$

where

$$\delta_f = \begin{cases} 4n^2 & \text{if } \Sigma \text{ is feasible} \\ 0 & \text{otherwise} \end{cases}$$

and

$$\mathrm{NL} = \text{Number of 1's in } \Sigma \tag{15}$$

The rational for assigning a high δ_f value to the feasible chromosomes is to steer the genetic process towards more and more feasible individuals. An infeasible chromosome might acquire a maximum fitness value as indicated in (5). The reason for assigning a $4n^2$ value to all feasible chromosomes is to keep at least a distance of 2n between feasible and infeasible chromosomes.

The addition of index NL ensures the fact that the 'best' individual in a population will be the most connected one; an organizational structure with more interactional links among its constituent DMUs will be considered 'stronger' by the algorithm than the rest. The index NL can be modified to reflect the designer's requirements on the interactional structure of an organization, i.e., a least connected structure can be considered the 'strongest' chromosome in a population.

4.1.5 Computation Of Solutions

This section presents a detailed description of the genetic process employed to solve the design problem.

Feasibility of Schema

As mentioned earlier that the user-defined constraints result in a schema which forms the building blocks for the generation of future population of chromosomes or structures. The genes (interactions) that are ruled in or ruled out by the designer remain constant throughout the evolution process. The genetic difference among several chromosomes is due to the optional genes (x's in the bit string.) Therefore, if there exist errors (violations of structural constraints) caused by the customary genes in the schema of an organization structure, they will be

propagated through out the entire population of chromosomes; the evolution process will never yield any feasible organization structure. In order to avoid this wastage of time and computation effort, the following checks are performed on the initial population of chromosomes prior to invoking the genetic process.

- The Kernel chromosome is checked for the constraint R2.
- The Kernel chromosome is checked for the constraint R3.
- The Kernel chromosome is checked for the constraint R4.

If the initial population fails any of these tests, the genetic process is immediately halted and designer is warned for the infeasibility of the schema. The first check is based on the rational that if an organizational structure with only the customary interactions among its DMUs lacks the acyclicity requirement, then acyclical structures can not be generated by adding more interactions to it. The same rational applies to the rest of the checks performed on the initial population. Note that even after these checks are performed on the Kernel chromosome, one can not guarantee the feasibility of the schema in terms of the structural constraint R1. The check for the feasibility of schema for R1 is a very involved process and therefore is dropped in favor of the speed of the process.

Crossover

In a genetic algorithm, crossover recombines the genetic material in two parent chromosomes to make two children. The children are made by cutting the parents at a (or some) point(s) and the chromosomal material is swapped between the cut point(s). In the methodology employed to generate DMO architectures, a 6-point crossover operator is used, which cuts the parents at six points, with one cut at each e, s, F, G, H, and C part of the chromosomes and the children are made by exchanging the parental genetic material after the cut. An example of this 6-point crossover is presented in Figure 4.1.4 during a run of the genetic algorithm. The children are made by cutting the parents at the points denoted by the vertical lines and exchanging genetic material after the cut. A Petri net representation of the process in Figure 4.1.4 is shown in Figure 4.1.5. It is obvious in Figure 4.1.5 that the crossover operator can produce children that are drastically different from their parents. Another important feature that the crossover will not introduce differences for a bit in a position where both parents have the same value; thus, preserving the customary interactions introduced by the designer.

Parent 1:	1	1	1	1	0	0	1	1	0	0	0	1
Parent 2:	1	0	1	0	0	0	0	0	0	0	0	0
Child 1:	1	0	1	0	0	0	1	0	0	0	0	0
Child 2:	1	1	1	1	0	0	0	1	0	0	0	1

Figure 4.1.4 6-Point Crossover

Mutation

When bit mutation is applied to a bit string it sweeps down the list of bits, replacing each by a randomly selected bit if a probability test is passed. The bit mutation operator employed for the methodology is a restricted one in the sense that it only operates on the optional links in a bit string representation of an organization structure, i.e., $\Sigma[2]$, $\Sigma[4]$, $\Sigma[7]$, $\Sigma[8]$, and $\Sigma[12]$ are the only bits in Σ in (6) where the mutation operator is applied. The probability parameter associated with the mutation operator is usually quite low. However, contrary to this practice, a large probability parameter used in the methodology for generating DIS architecture was found more promising in generating feasible solutions than its lower counterpart. The genetic algorithm with an unusually high rate of mutation generated feasible structures faster than the algorithms with very low mutation rates. Figure 4.1.6 shows an example of bit mutation on a Petri net representation of structures.

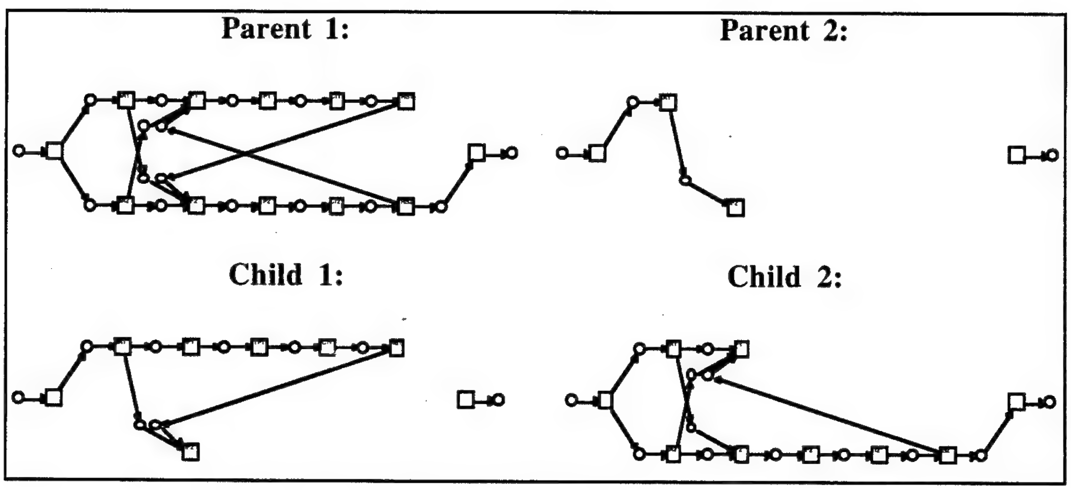

Figure 4.1.5 Petri net Representation of 6-Point Crossover

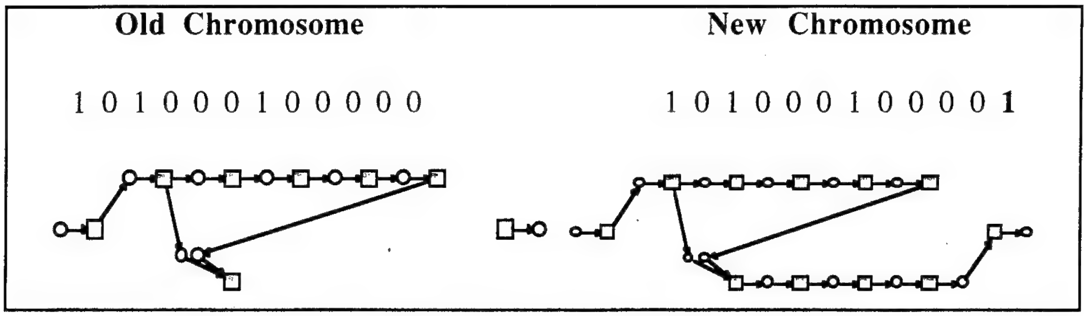

Figure 4.1.6 An Example of Bit Mutation

Selection

The genetic process applies mutation and crossover operations to the individuals of a population to yield a new generation (population) of chromosomes. At present, the algorithm for generating DIS architectures selects the 'stronger' individuals in a population to mate and reproduce. The strength of an individual is determined by the fitness value (15) assigned to it by the evaluation function. The *elitist* (Heckerman, 1990) approach is employed to add chromosomes to the subsequent populations.

Elitism

As can be noted that the best member of the population may fail to produce offspring in the next generation. The elitist strategy fixes this potential source of loss by copying the best member of each generation into the succeeding generation. In the present approach all those feasible members of a previous population are added to the new generation that are stronger than the members of the new generation.

4.1.6 Application

The approach presented in this paper was applied to the design problem illustrated by Zaidi and Levis (1995) in their approach of generating large-scale DIS architectures through a hierarchical arrangement of the system. Due to the computational and memory intensive nature of the design problem, the implementation of the Lattice algorithm by Remy and Levis (1988) could not handle this same problem. The user-defined constraints for this design problem are given as follows.

After generating the initial population—Universal and Kernel chromosomes—the Kernel chromosome is checked for the infeasibility of the schema. Once all checks are cleared, the genetic process gets started. The process can be stopped as earlier as the first feasible chromosome is generated, or whenever the time limit is surpassed. The methodology presented in this paper is implemented on *DesignCPN*™ (1991), a commercially available software for

Colored Petri net modeling, by program codes written in ML^{TM} with the logic of the process implemented by a Colored Petri net model. An ML program, in the end, can be used to transform or decode the bit string representation of an organization structure to its Petri net representation. The methodology is applied to the illustrative example: the algorithm yielded the first feasible chromosome in the 4th population, Figure 4.1.7 shows the Petri net representation of this structure; overall 30 populations were generated before the process was terminated by the user, the best structure in the 30th population is shown in Figure 4.1.8 in terms of its Petri net representation.

$$\mathbf{e} = \begin{bmatrix} 1 & x & 1 & 1 & 1 & x \end{bmatrix} \qquad \mathbf{s} = \begin{bmatrix} 0 & x & 1 & 1 & x & 1 \end{bmatrix}$$

$$\Sigma: \quad \mathbf{F} = \begin{bmatrix} \# & 1 & x & x & 0 & 0 \\ x & \# & x & x & 0 & 0 \\ x & x & \# & x & x & x \\ x & x & x & \# & x & x \\ x & x & 0 & 0 & \# & x \\ x & x & 0 & 0 & 0 & \# \end{bmatrix} \qquad \mathbf{G} = \begin{bmatrix} \# & 0 & 0 & 0 & 0 & 0 \\ 0 & \# & 0 & 0 & 0 & 0 \\ 0 & 0 & \# & 0 & 0 & 0 \\ 0 & 0 & 0 & \# & 0 & 0 \\ 0 & 0 & 0 & 0 & \# & x \\ 0 & 0 & 0 & 0 & x & \# \end{bmatrix}$$

$$\mathbf{H} = \begin{bmatrix} \# & x & 0 & 0 & 0 & 0 \\ x & \# & 0 & 0 & 0 & 0 \\ 0 & 0 & \# & x & 0 & 0 \\ 0 & 0 & x & \# & 0 & 0 \\ 0 & 0 & 0 & 0 & \# & 1 \\ 0 & 0 & 0 & 0 & 0 & \# \end{bmatrix} \qquad \mathbf{C} = \begin{bmatrix} \# & 0 & 0 & 0 & 0 & 0 \\ 0 & \# & 0 & 0 & 0 & 0 \\ x & x & \# & 1 & x & x \\ x & x & 0 & \# & x & x \\ 0 & 0 & 0 & 0 & \# & 0 \\ 0 & 0 & 0 & 0 & 0 & \# \end{bmatrix}$$

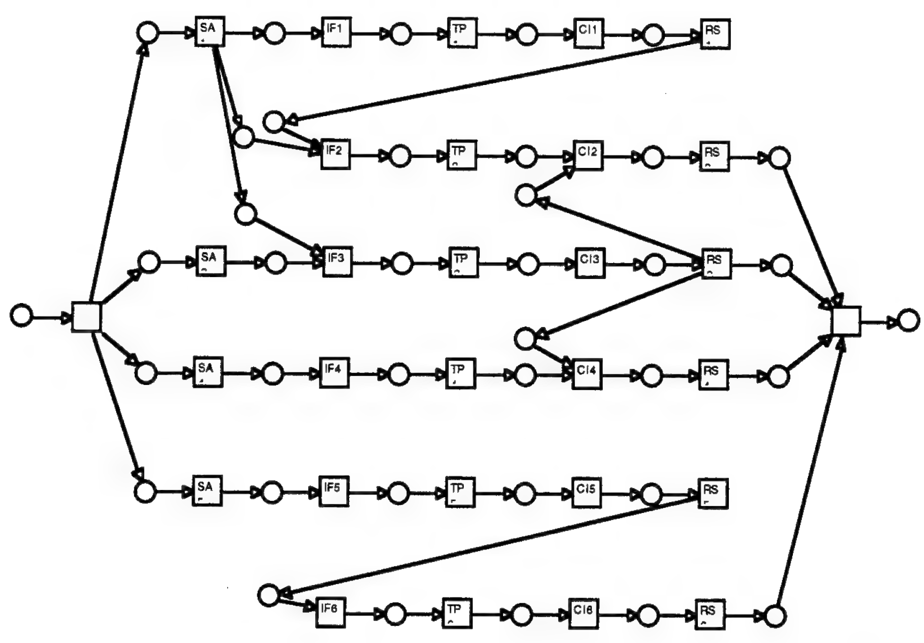

Figure 4.1.7 First Feasible Structure Generated by the Methodology

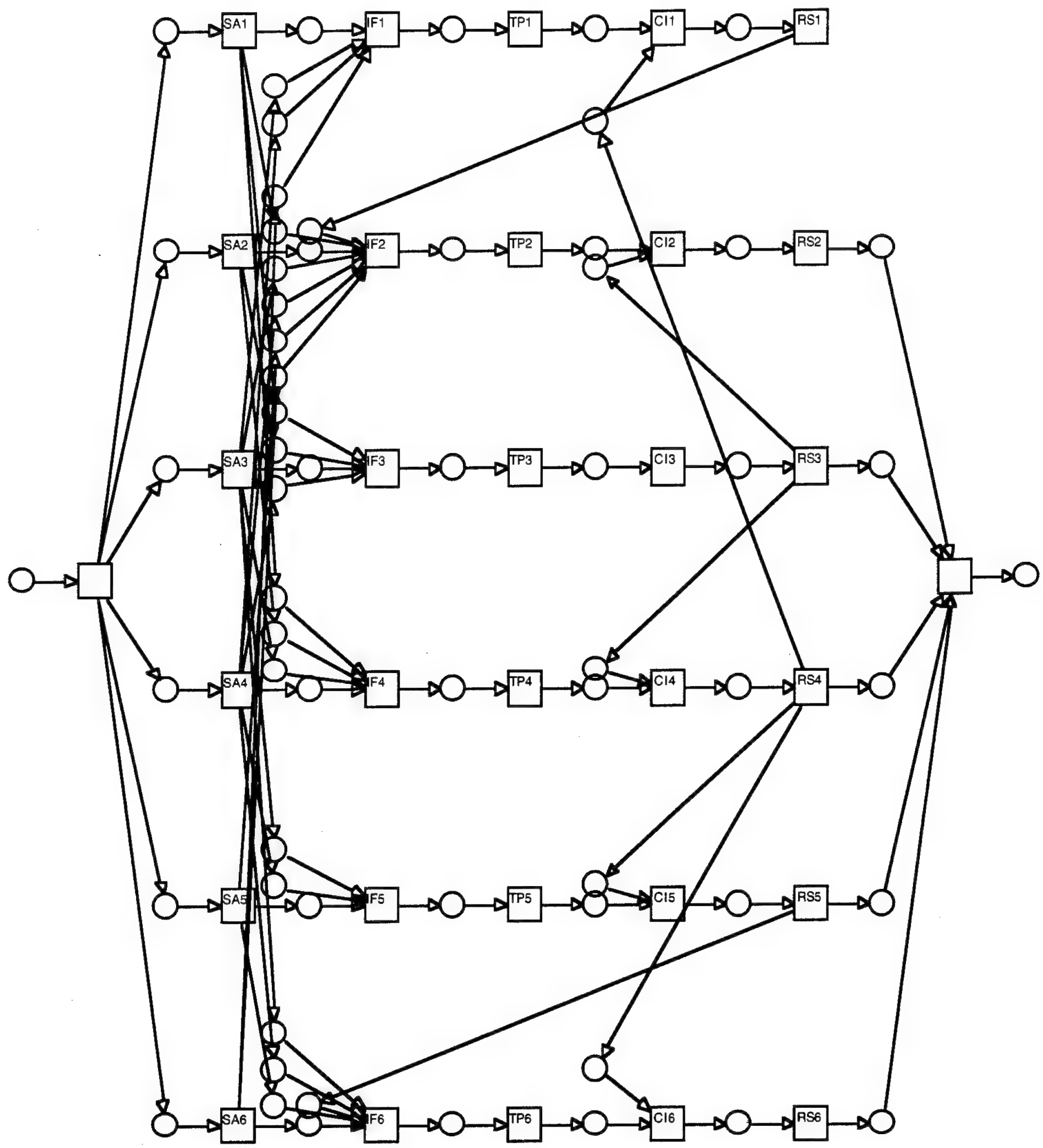

Figure 4.1.8 The Best Structure in the 30th Generation

The structure in Figure 4.1.7 is important in the sense that it was eliminated from the solution space by the methodology in Zaidi and Levis (1995) because of the design decisions taken at an earlier stage by the designer - a requirement of that methodology. The results of the performance of the methodology for the illustrative example are summarized in Figures 4.1.9, 4.1.10, and 4.1.11.

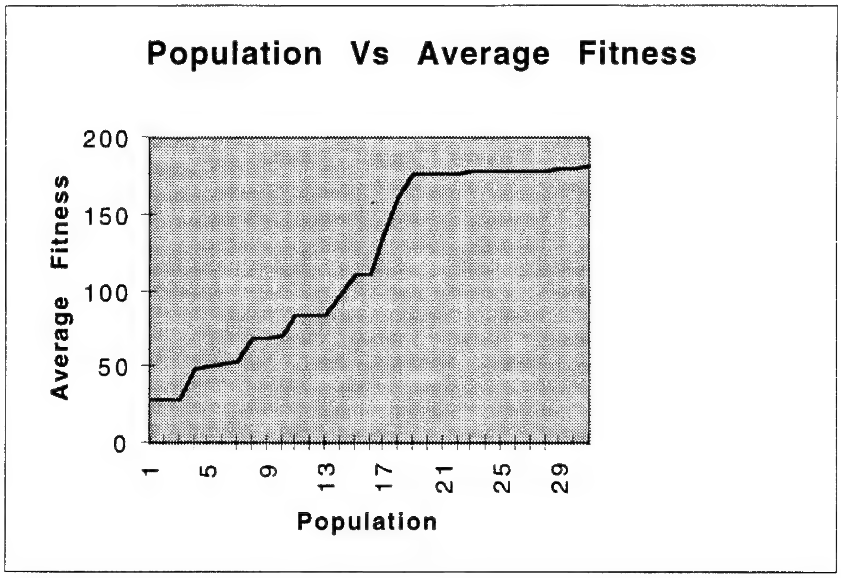

Figure 4.1.9 Population vs. Average Fitness

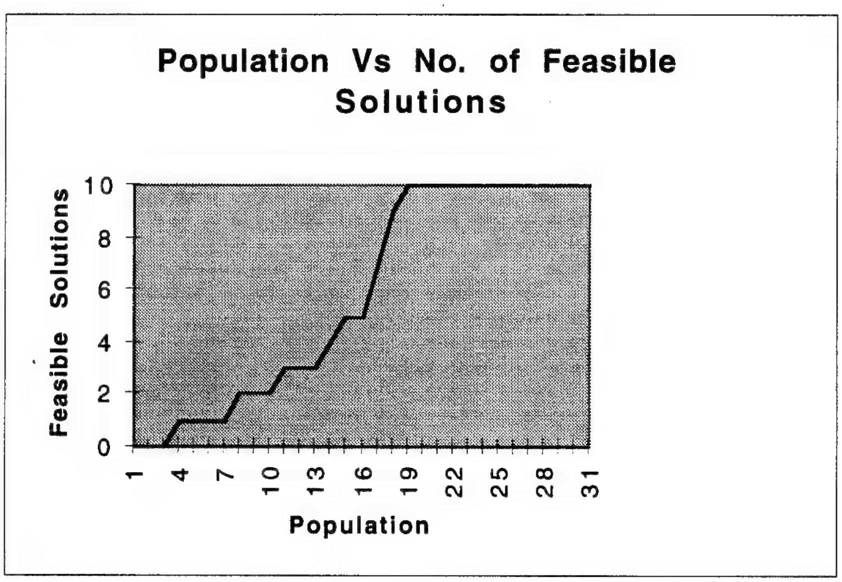

Figure 4.1.10 Population vs. No. of Feasible Solutions

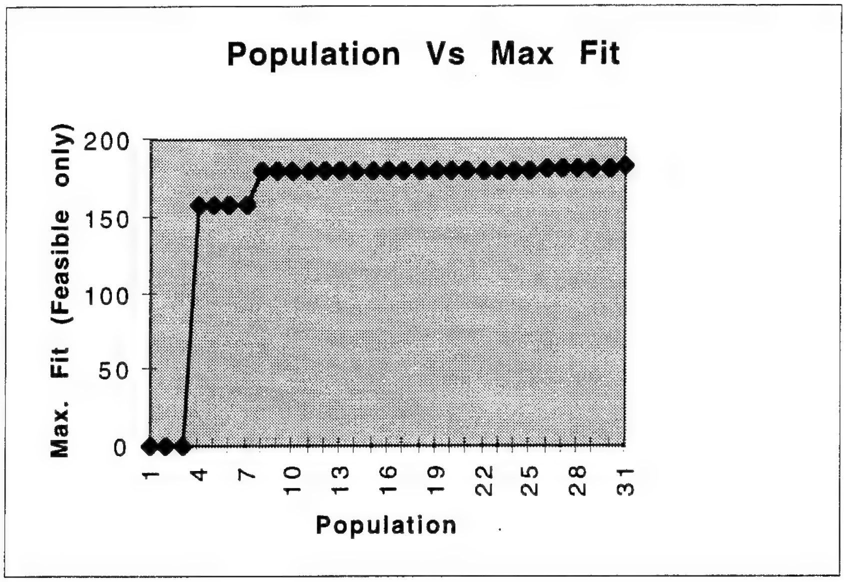

Figure 4.1.11 Population vs. Max Fit

4.1.7 Conclusions

A methodology for generating DMO architectures using genetic algorithm has been developed. The approach provides an alternative for the generation of large-scale organizational structures where the combinatorial nature of the problem makes the previous approaches computationally expensive and infeasible. Another advantage of using the approach is that the additional structural and performance criteria can be made an integral part of the design algorithm to direct the search for the solution in a particular direction.

4.2 DISTRIBUTED COORDINATION IN ADAPTIVE COMMAND AND CONTROL TEAMS (Perdu and Levis)

4.2.1 Problem Definition

A team may be defined as a group of experts, with overlapping areas of expertise, that work cooperatively to solve decision problems. Command and Control teams are a class of teams where:

- A specific set of tasks is assigned to each team member;
- The team members are well trained for the tasks they are supposed to carry out;
- The team members have the common goal to solve satisfactorily the decision problem;

- The decision making process is subject to strong time constraints;
- The decision is made in a very uncertain environment; and
- Inappropriate decisions may lead to catastrophic consequences.

Teams are set up for the performance of specific missions. For the execution of a mission, different functions are allocated to the team members according to their area of expertise; the organization exhibits a pattern of interactions between team members. Coordination between team members to ensure the coherence of the distributed processes is critical for the effective operation of a team. Moreover, since teams face a variety of different missions or problems and since no single organization is optimal for all situations, the team organization needs to be adaptive, that is dynamically reconfigurable to meet changing demands. Here again, coordination between team members is critical, if smooth switching from one configuration to another is to occur.

Coordination between team members is subject to the constraint that the performance of the organization has to remain within acceptable limits or, in other words, that performance requirements are still satisfied. The violation of one or several performance requirements can be the trigger for setting up a new coordination scheme. Measures of Performance of the organization need to be evaluated so that the coordination scheme can be qualified as acceptable or unacceptable. Section 4.2.3 describes how decision making organizations can be evaluated.

Coordination can be performed in a centralized way: a central controller can keep track of the state of the system and can adapt the configuration according to the state and the demands from the environment. In Command and Control teams, because of the uncertainty of the environment they are dealing with, and because of the strong time constraints to make decisions, the coordination has to be distributed and user-initiated. In a normal mode of operations, team members have to interact to solve together the problem resulting from the uncertainty of the environment and thus perform the mission in an effective manner. To switch from one configuration to another, team members have to collaborate to identify the need for change, to select the new configuration, and to smoothly implement the new selected configuration. This distributed coordination is the focus of this task. The problem addressed is: Can distributed coordination be implemented in Command and Control teams at the design stage?

Studies on team decision making (Cannon-Bower et al., 1990; Orasanu and Salas, 1993) have shown that teams with a history of working together perform better than teams in which team members are not used to work together . The former teams seem to coordinate better than the latter. In the context of Command and Control teams, where there is a high turnover of personnel, this characteristic is highly undesirable. One way of mitigating the problem and reducing this variation in performance is to approach the problem of coordination at the design stage of an adaptive team. The allocation of the tasks to the team members and the definition of the interactions between them should be done so that each team member knows what to do only on the basis of the information he has direct access to. Distributed coordination will thus take the form of coordination rules used locally by the different team members that will define what function to perform under what circumstance. A methodology for adaptive team design is needed. Based on Colored Petri Net, this methodology will allow (1) to derive the coordination rules by analysis of the structural properties of the Colored Petri Nets representation of the team operations, (2) to validate and verify these coordination rules, and (3) to predict the performance of the team by simulation of the Colored Petri Net. More details on the proposed approach are given in Section 4.2.4.

This section is organized as follows. The section 4.2.2 attempts to clarify the concept of teams and stresses out the need for coordination. Section 4.2.3 presents the modeling approach, based on Petri nets, to represent fixed and variable structures decision making organizations. Section 4.2.4 discusses the approach to the design problem.

4.2.2 The Concept Of Teams

Definition

Salas et al. (1992) defined a team "as a distinguishable set of two or more people who interact, dynamically, interdependently, and adaptively toward a common and valued goal/objective/ mission, who have each been assigned specific roles or functions to perform, and who have a limited life-span of membership." A team is set up to accomplish a certain mission, the overarching common goal of the team members being the completion of the mission. The central point of the definition is that task completion requires: (a) a dynamic exchange of information and resources among team members, (b) coordination of task activities (e.g., active communication, back-up behaviors), (c) constant adjustments to task demands, and (d) some organizational structuring of team members.

In the "Economic Theory of Teams" (Marschak and Radner, 1972), the definition of a team introduces more precisely the concept of information. Actions performed in an organization differ, in general, from those of a single person in two respects:

- the kind of information on the basis of which each organization member decides about his actions may differ from one member to another.
- the interests and beliefs of each member of the organization may differ from the interests and beliefs of his fellow members.

A team is defined as an organization in which the first, but not the second characteristic is present: the team members have the same interests and beliefs but do not share the same information. Therefore, a team is also characterized by more than one information source.

Teams perform under various decision structures that require different levels of interaction and communication. The process carried out by a team may include several decision making processes (Vroom and Yetton, 1973):

- Autocratic decision making: The team leader obtains all necessary information from team members, then decides on the solution alone without sharing the problem;
- Consultative decision making: The team leader shares the problem with the team members and then gathers ideas and suggestions before making the decision alone;
- Participative decision making: The team members share the problem and the generation and evaluation of alternatives in order to reach mutual agreement on the solution.

As pointed out by Duffy (1992), participative decision making is the most closely aligned with the concept of team decision making, but all of the above strategies are used by teams at various times, depending on the task requirements, time pressure and the need for acceptance of the decision. Command and Control Teams are expected to use the two first strategies, autocratic and consultative decision making. These teams are built around a commander, who is responsible for making the organization's decision. The other team members' roles are support roles. The first role is to collect information about the environment, filter and fuse data coming from a variety of origins in order to construct the tactical picture, that is, a description of the situation, on which the commander will make his decision. The second support role is to implement the commander's decision. Depending on the situation, the commander can delegate some of his responsibilities to his subalterns and/or ask them for advice, but remains responsible for the final decision.

Coordination

Team performance is not only a function of the potential of individual team members to perform their assigned tasks, but also a function of the ability of team members to coordinate

their work flow and communicate effectively with one another. Individual skills and knowledge are not sufficient for successful team performance. The processes performed by the different team members are interdependent, meaning that each member accomplishes a part of the overall team mission. Each individual contribution needs to be merged to produce the final team product subject to constraints of tasks dependency (the output of some tasks is the input to other tasks) and of timing. Coordination is required to accomplish this in an effective manner and communication is central to team activity because individual resources must be appropriately utilized through interaction processes.

Related to effective coordination is the concept of shared/mutual mental models. In formulating his "Expert Team of Experts" methodology, Athans (1982) argued that an effective C2 organization is more than a team of individual experts: in addition to his own expertise, each commander needs to develop a mental picture or model, which represents an aggregated version of the tactical decision making process of his fellow commanders with whom he must coordinate, interact and compete for scarce resources. The concept of shared/mutual mental models has also emerged from recent investigations of team decision making in natural environment (Cannon-Bower et al., 1990; Orasanu and Salas, 1993). While not a fully developed theory, this concept accounts for certain team phenomena. Team members have shared mental models when they possess an accurate and equally detailed understandings and conceptualizations (i.e., mental model) of the various requirements of team functioning and performance. The greater the accuracy and overlap among team members, the greater the likelihood that the team members will predict, adapt and coordinate with one another successfully even under stressful or novel conditions. Teams, with a history of working together, interact differently than ad hoc teams. Experience of working together leads members to build a shared mental model for the teams, which allows members to predict other's behavior and needs. This team member predictability leads to better team coordination and performance. Using a simulated military command and control scenario, Kleinman and Serfaty (1989) examined the effects of workload, overlap of functional responsibility, and distribution of knowledge resources on communications and performance. As the tempo of tasks increased from low to moderate, explicit verbal coordination increased. But as workload became high, the amount of communication went down. Virtually all resource transfers were unsolicited, resulting in implicit coordination. Kleinman and Serfaty interpreted this pattern as exercise of shared mental models that allowed participants to anticipate each other's resource needs and actions.

Adaptive Teams

Survivability, flexibility and reconfigurability are three properties that are sought for in adaptive teams. A team is survivable when it can still perform its missions when failures occur. Flexibility means that a team may adapt to the tasks it has to process, or to their relative frequency. Reconfigurability means that it can adapt to changes in its resources or in its missions.These three properties obviously overlap. To address these three properties, Monguillet (1988) defined three types of variability in organizations:

- Type 1 variable: if the organization adapts to the input it processes. Some patterns of interactions between team members may be more suitable for the processing of a given input than others.
- Type 2 variable: if it adapts to changes in the environment. The performance of a team may depend strongly on the state of the environment. A team with a given structure may accomplish its mission perfectly if the arrival rate of inputs is low. This structure may not be optimal if the arrival rate is high. Responsibilities and relationships between team members may need to change.
- Type 3 variable: if it adapts to changes in the system's parameters. In case of failure or unavailability of some team members, for example, it might not be possible to accomplish the mission with the current structure. The structure needs to be changed.

An organization often exhibits these three types of variability simultaneously.

Back-up schemes need to be implemented for an adaptive team to be able to face changing demands. Set up at the team design stage, these back-up schemes require to implement some degree of redundancy for the processes performed by the team members and for the information exchanged between them. In addition to the task allocation to team members, coordination rules need to be derived. In the case of Command and Control teams, this process of coordination needs to be distributed as much as possible because of the strong time constraints of the environment the team is dealing with. These coordination rules must allow the team members to decide who does what and under what circumstances based on the information they have access to. Through interactions, each team member can increase its individual perception of the environment but the problem is all the more complex as this information is noisy because of the uncertainty of the environment. Prior to the use of the coordination rules, a distributed belief revision process has to take place to assess the need for change and the selection of the most appropriate configuration. These problems are the focus of this task.

4.2.3 Petri Net Modeling And Evaluation Of Teams

Petri Nets (Peterson, 1981; Reisig, 1985) and Colored Petri Nets (Jensen, 1987) have been found to be very convenient for describing the concurrent and asynchronous characteristics of the processing of information in a decision making organization. This section presents the methodology to model and evaluate decision making organizations/teams.

Model of the Interacting Decision Maker

Boettcher and Levis (1982) have defined an ordinary Petri Net model of an interacting decision maker. This model has been converted to a Colored Petri Nets by Levis (1993). It consists of five stages, as shown in Figure 4.2.1. In the Situation Assessment (SA) stage, the decision maker receives an input from the environment, processes this information and produces an output that can be transmitted to other members of the organization. In the Information Fusion (IF) stage, the decision maker can merge his own situation assessment with some other information received from other members of the organization. The decision process takes place in the Task Processing (TP) stage. In the Command Interpretation (CI) stage, the decision maker can receive command from decision makers hierarchically superior to him that can constraint his production of a response in the Response Selection (RS) Stage.

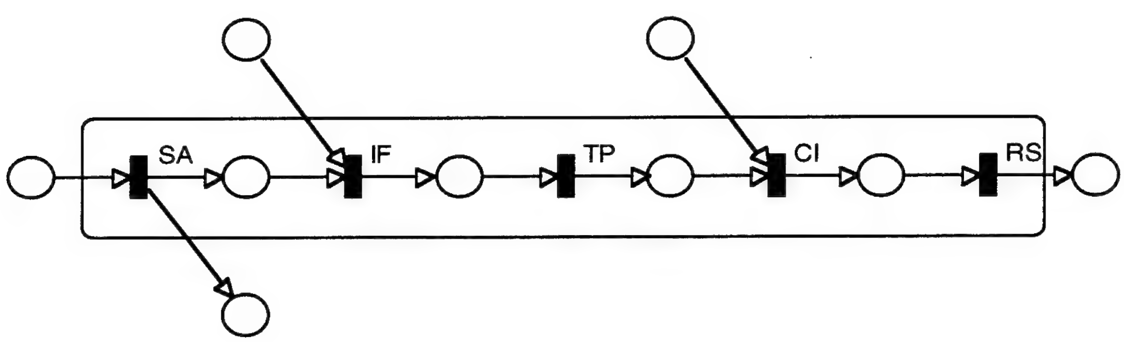

Figure 4.2.1 Petri Net model of the Interacting Decision Maker

Model of Decision Making Organizations

A Petri Net model of a decision making organization is then constructed by connecting appropriately different basic decision maker models. Remy (1986) has defined a procedure using the analytical properties of Petri Nets and lattice theory to generate the set of organizations satisfying a set of connection and structural constraints. The solution architectures are numerous but the interest of Remy's approach is that the large set of feasible

solutions can be characterized by the minimally and maximally connected organizations. Instead of looking at the set of solutions one can focus only on its boundaries.

Let us look at an example from Remy and Levis (1988): the Warfare Commander problem. This organization has five decision makers. DM_1 and DM_2 act as the sensors of the organization (Sonar Operator (SO) and Radar Operator (RO) for example). They both receive information from the external environment (threat detection). They may or may not share this information. However, DM_1 has to send his revised assessments to DM_3 which acts as the Executive Coordinator (EXCO). DM_4, the Anti-Air Warfare Commander (AAWC) and DM_5, the Anti-Submarine Warfare Commander (ASWC) produce the organization's response (firing of missiles for AAWC or torpedoes and depth charges for ASWC). They receive orders from the coordinator DM_3 and receive information from DM_1 and DM_2. They may also share their results. The model displayed on Figure 4.2.2 is one of the maximally-connected candidate architectures. DM_1 and DM_2 share their initial assessment as represented by the interaction places connecting the transitions SA1 to IF2 and SA2 to IF1. DM_1 and DM_2 send their revised assessment to the three other members of the organization (interaction places connecting RS1 and RS2 to IF3, IF4 and IF5). DM_3 generates orders that are sent to DM_4 and DM_5 as represented by the interaction places connecting RS3 to CI4 and CI5. Finally in the case of this particular solution, DM_5 has to send its selected course of action to DM_4, as represented by the interaction place from RS5 to IF4.

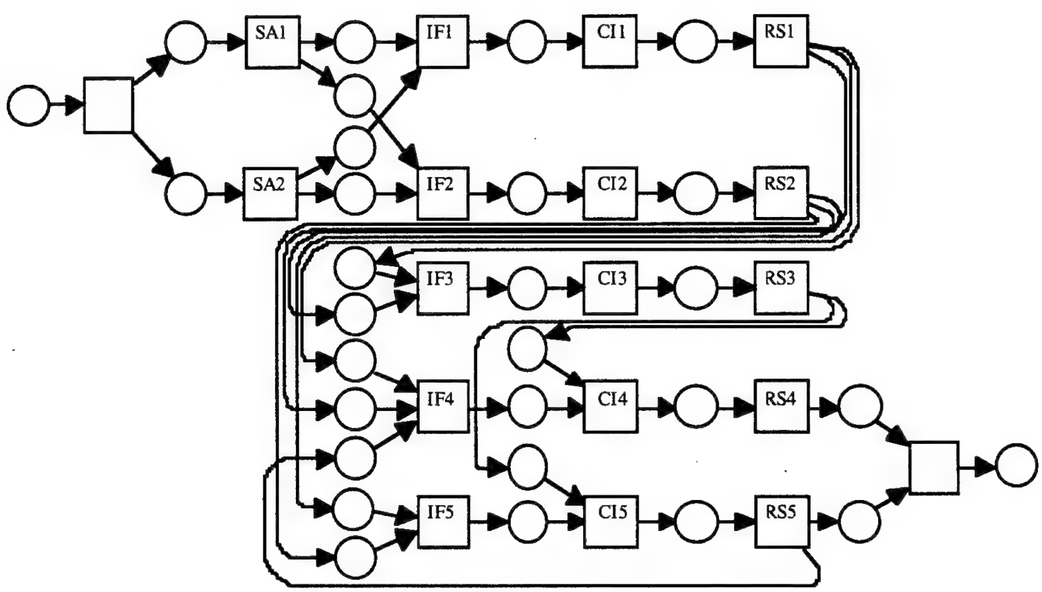

Figure 4.2.2 Example of a Decision Making Organization

Evaluation of Decision Making Organizations

An organization/team can be seen as a system performing a task in order to achieve a mission. The extent to which it does that is assessed by using the formalism of the System Effectiveness Analysis (SEA) methodology (Martin and Levis, 1987). In a first step, Measures of Performance are evaluated. In a second step, Measures of Effectiveness that measure the extent to which the Measures of Performance satisfy the performance requirements are derived.

Different Measures of Performance can be evaluated to assess the effectiveness of an organization. The main Measures of Performance are Response Time, Throughput Rate, Accuracy, and Human Workload for which the definitions are described below. The computation of these measures of performance requires the consideration of a scenario which is a set of inputs to the organization satisfying a certain probability distribution.

Response Time: The response time of an organization is critical for C3I applications and evaluates the ability of the organization to produce an output in a specific window of opportunity. The response time of an organization is the easiest thing to compute in the Petri Net framework. The time delay equation of an organization can be computed without any simulation. However, when queuing occurs in specific part of the organization, simulation is required to assess the effect of the resources availability on the response time.

Throughput Rate: Related to the response time and the availability of resources, the throughput rate measures the number of inputs the organization can process in a certain amount of time. It is related to the rate at which it produces outputs. Queuing can occur in some part of the organization and will affect the throughput rate.

Accuracy: Accuracy measures the ability of an organization to produce the right output in an uncertain environment. It depends on the type of algorithms used to deal with the uncertainty of the input. The accuracy measure is obtained by comparing the output of the organization to the "ground truth", which is the output the organization should produce given the input. A cost is associated for discrepancies between the organization output and the "ground truth". For example, identifying a friend as a neutral is less costly than identifying a foe for a friend. A cost matrix has to be derived in which the cells define the cost for the discrepancy between the organization output and the ground truth. Therefore, the scenarios used to evaluate the accuracy of the organization should include for each input, in addition to the probability of occurring, the output that the organization should produce. For a given scenario, the Accuracy measure is

then equal to the expected cost and has to be computed for every set of algorithms being used inside by the decision maker. The computation of the accuracy measure is infeasible in the Ordinary Petri Nets framework but can be done in Colored Petri nets.

Decision Maker Workload: Boettcher and Levis (1982) developed an evaluation of the Human Decision Maker workload based on Information Theory. Using entropy H, as the measure of the uncertainty in a random variable w_i that takes values w_{ij} and where:

$$H(w_i) = - \sum_j p(w_{ij}) \log_2 p(w_{ij})$$

the total information processing activity of a decision maker, G may be expressed as:

$$G = \sum_i H(w_i)$$

where w_i's are all the variables handled by the Decision Maker in his algorithms executed to carry out his tasks.

Coordination and Variable Structures

In the context of the Petri net representation of fixed decision making organizations, Grevet (1988) has addressed the coordination problems in two classes of issue:

- The synchronization of the activity during the decision making process
- The consistency of the information processed by the different members of the organization.

In this work, an attempt was made to derive coordination measures

The investigation of variable structure decision making organizations was initiated by Monguillet (1988). He extended the System Effectiveness Analysis methodology to account for variable structures by using Predicate Transition Nets (Genrich, 1987). Demaël (1989) extended the design methodology of Remy (1986) to Type-1 variable decision making organizations, i.e., organizations that adapt their structures to the inputs they process. More recently, Lu (1992) addressed the problem of coordination in variable decision making organization by decoupling the organization into two layers: (1) the system layer which consists of the set of physical entities with their interconnectivities and (2) the coordination layer which describes the rules of operation of the system layer. He defined a methodology, based on Colored Petri Nets, to develop the coordination layer given the system layer so that an

organization can have a variable structure that exhibits the three types of variability of Monguillet. A set of fixed structure is designed for each input situation and then folded into a set of realizable variable structures called coordination schemes. The organization can switch from one coordination scheme to another by implementing the supervisory layer within the coordination layer. The supervisory layer changes the coordination strategy according to information about the system and environment. In his work, Lu addressed one of the key aspects of designing variable structure organizations, the coordination constraint: when folding fixed structures into a variable structure, a component must have the necessary information to make different responses, i.e., if the inputs are the same for two input situations, the component should not be able to produce different responses for these two input situations.

4.2.4 Approach To Solve The Problem

Distributed coordination rules need to be defined at the team design phase. Coordination for reconfiguration is not or weakly addressed in current Systems Engineering design methodologies applied to distributed human decision making. At most, they recognize the need to allocate a function to several physical components to ensure some degree of redundancy and back ups, but fail to define precisely the conditions under which back up has to take place. This is due to the facts that these methodologies provide only a static representation of the system or team structure, and that adaptivity and coordination are dynamic in nature. These problems are addressed much later either during training or during operations. This is often through informal interactions between team members (what constitutes the "glue" of the system) that adaptivity can take place. The proposed approach to address these problems is as follows. In a first step, a methodology for the design of adaptive teams has to be developed. Then, a procedure to derive coordination rules from the design has to be specified. Finally, these derived coordination rules needs to be validated. Let us examine in more details these different steps.

Design of Adaptive Teams

The aim of the design of adaptive teams is to define the responsibilities of each team member, that is to allocate the functions to the different decision makers, for the different modes of the team operations.

The first step is to generate the *Operational Concept* that will drive the design. The Operational Concept specifies what the team is supposed to do, the type of tasks it will carry out, the missions it will execute and how it will do them. An Operational Concept corresponds to the broadest requirements.

The second step of the approach is to perform a *Functional Decomposition*. Functions necessary for the execution of the missions defined in the Operational Concept are identified and further decomposed into subfunctions that need to be performed for the execution of the function. This decomposition process can then be applied further to the subfunctions until each subfunction correspond to an elementary task that can be allocated to a single team member. The *Functional Architecture* is derived by specifying the data exchanged between subfunctions. It can be represented by a Petri net that shows how the functions interact for the execution of a mission. An example is shown on Figure 4.2.3. Inputs from the environment are processed by functions f1 and f2. The output of f1 is processed by function f3, while f4 needs the output of both f1 and f2 to be performed. Function f5 needs the results of functions f3 and f4 to produce the team output.

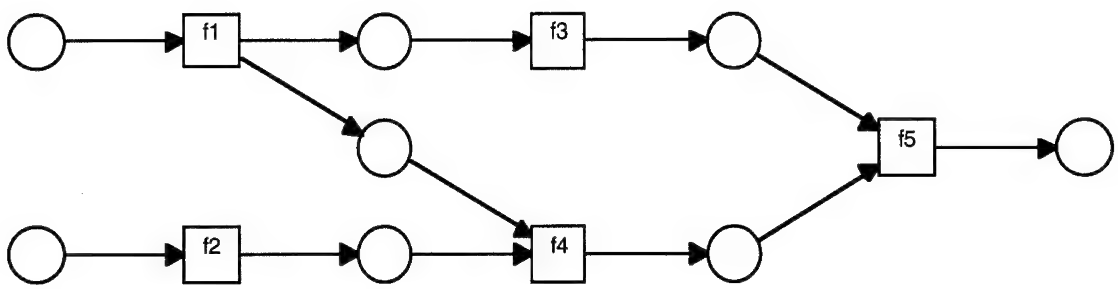

Figure 4.2.3 Petri Net Representation of a Functional Architecture

A functional Architecture can be constructed for each mission and for each input pattern. The obtained Petri Nets need then to be folded together into a single structure that depicts all the functions and interactions necessary for the execution of the set of missions. The functions used for different missions are superimposed. The resulting structure is a variable one and is represented by a Colored Petri Net.

In the last step, functions are allocated to team members subject to the limited processing capabilities of each member, and to the limited capacity of the communication network connecting the different team members. The derived variable structure decision making organization is of types 1 and 2: it adapts its responsibility distribution and its interaction patterns to the inputs it has to process and to the missions it has to perform. To obtain type 3 variable structures, there is a need to take into account redundancy of processes and to define some back-up strategies. Two basic types of back-ups can be considered: vertical back-up and horizontal back-up.

Vertical Backup: Vertical back-up is the transfer or devolution of responsibilities from one team member to another when the former is unable to perform the function(s) he is responsible for. Vertical back-up is implemented by defining different allocation of functions to the team members for different modes of operations. Figure 4.2.4 shows different function allocations to team members for the functional architecture of Figure 4.2.3. The upper part of the Figure corresponds to the distribution of responsibilities in a normal mode of operations: DM1 is responsible for function f1, DM2 for function f2, DM3 for function f3, DM4 for function f4 and DM5 for function f5. The lower part is the distribution of responsibilities in a vertical back up mode when DM5 is unable to perform f5 which is then performed by DM4.

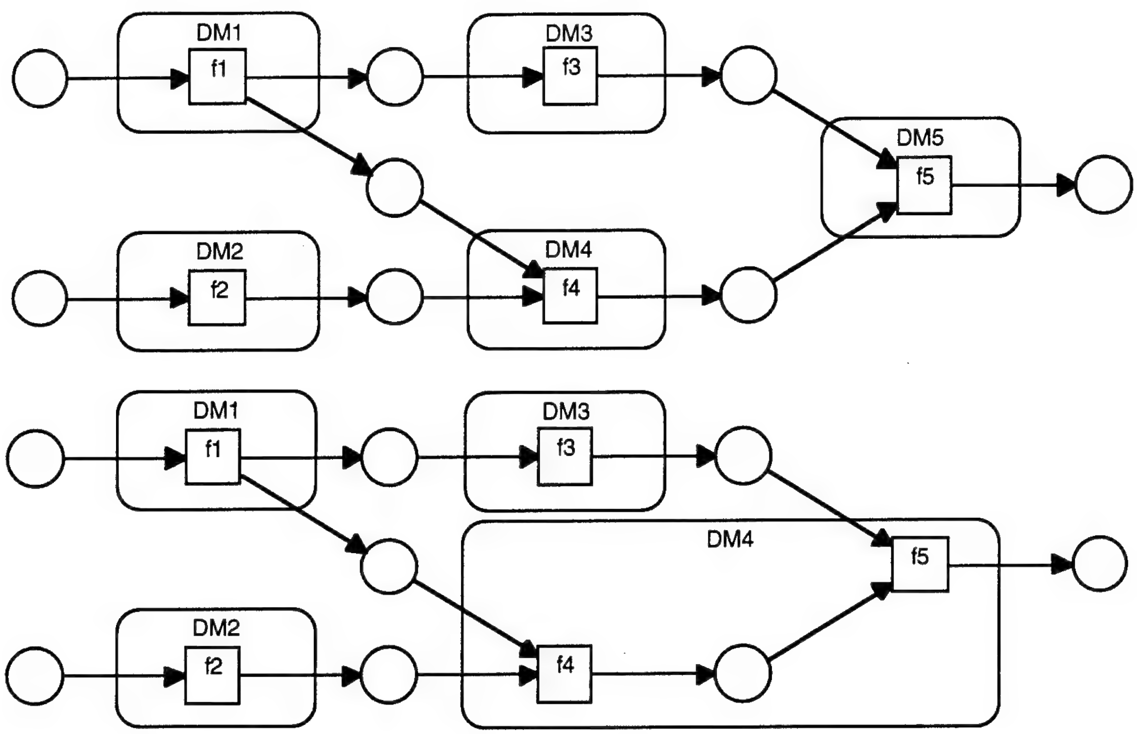

Figure 4.2.4 Different Function Allocations for Implementation of Vertical Back-up

A representation of the team operations is then obtained by folding together the representations of different function allocations. Figure 4.2.5 displays the representation of the team operations obtained by folding together the two function allocations of Figure 4.2.4.

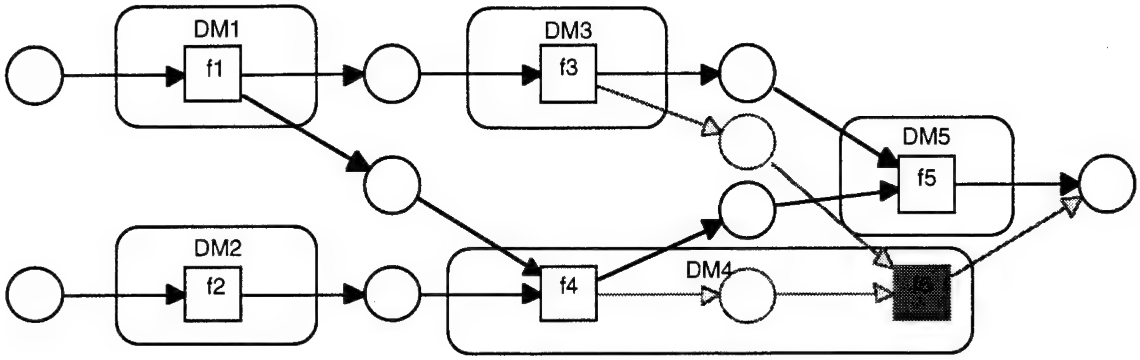

Figure 4.2.5 Representation of the Team Operations
Accounting for the Vertical Backup of Function f5 from DM5 to DM4.

Horizontal Backup: Horizontal back-up applies to functions that are performed or shared by several team members. When, because of the large number of inputs to process, a function can not be performed by a single decision maker, it can be allocated to several ones so that the overall task is shared between them. Horizontal back-up is therefore an extension of the share of the task of one or several team members to compensate for a team member's inability to complete his share satisfactorily. This extension of responsibility is used for load balancing between team members performing the same function. A common way to share responsibilities between team members is to assign geographical areas of responsibilities to each of them: a decision maker is responsible for processing all the inputs in his area of responsibilities. These areas of responsibilities can overlap and the team members need to coordinate to decide which input in the common area they each process. Horizontal back-up is implemented by adapting the size of the areas of responsibilities. In Figure 4.2.6, the functional architecture of Figure 4.2.3 has been modified so that the load induced by the execution of function f4 is shared between DM4 and DM5 that have been assigned different but overlapping areas of responsibilities. DM1 and DM2 know which input to send to each of them. The inputs in the common area of responsibilities are sent to both DM4 and DM5 who need to coordinate, as represented by the places connecting DM4 and DM5.

Redundancy in the team operations is therefore introduced by considering a mixture of vertical and horizontal back-ups. The result is the definition of a team whose members have overlapping areas of expertise. The team survivability is then related to the extent to which the team missions can still be carried out when some parts of the teams become unavailable.

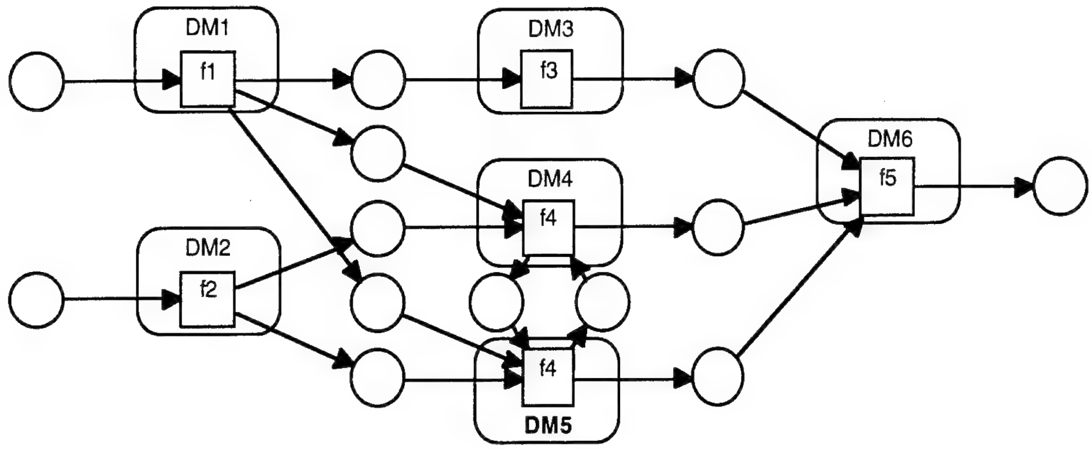

Figure 4.2.6 Horizontal Back-up

Derivation of Coordination Rules

The need for coordination rules is critical. For example, for vertical back-up, in Figure 4.2.5, DM4 needs to know when to perform f4 and DM3 needs to know to whom to send the output of f3. For horizontal back-up, in Figure 4.2.6, DM4 and DM5 need some criteria to adjust their areas of responsibilities and in case of changes, DM1 and DM2 need to know which input to send where. The goal is therefore to derive from the different possible modes of operations a set of coordination rules for each team member that specifies under what conditions:

- to perform some functions;
- to send the function output to a specific addressee;
- to identify and implement changes locally (that is with team members he is in contact with).

These changes should be seen as perturbations from a current/nominal mode. There is no need to reallocate completely the functions to the different team members. For example in Figure 4.2.5, the fact that DM4 can exercise vertical back-up to DM5 for the execution of function f5 does not affect at all DM1 and DM2. The second aspect is that if several back-up modes are possible to account for a particular problem, the team members should not face the conflict of choice between these several modes. The conflict needs to be solved at the design stage. To address these two aspects a graph-theoretic representation of transitions between modes of operations that will look like a lattice and for which conflicts will be solved is proposed. Each team member will have a different perception of this graph because of the locality of the changes: a transition graph for each team member will be derived from the transition graph for the team by considering only the changes in which the team member is involved in.

To derive the rules to go from one state to another, a method based on the analysis of the Colored Petri Nets invariants of each mode of operations and the integer programming approach to represent logical statements applied to these invariants will be developed.

Validation of Coordination Rules

Once the coordination rules have been derived, there is a need to check that they make sense and are realizable. This validation of coordination rules can be done in two steps: a static validation to identify the logical inconsistencies and then a dynamic validation to identify behavior problems and to evaluate the team performance.

Static Validation: Since the rules are defined locally, there is a need to check and maintain the correctness of the distributed coordination rule base. My intent is to use the approach for validation and verification of decision making rules developed by Zaidi (1994). He developed a Petri Net based approach to identify:

- Redundant rules
- Inconsistent/conflicting rules
- Circular rules
- Incomplete rules

The use of this method will allow to identify the logical problems that can impede the coordination process of adaptive teams when a decentralized approach is used.

Dynamic Validation: The dynamic validation will be done by simulating a Colored Petri Net representation of the team operations. There are two main objectives in dynamic validation. The first one is to study the transient state behavior of an adaptive team. The simulation will allow to identify problems associated with the switching from one mode to another. One can expect that in some cases divergence can occur: a team member providing back-up to another one can become overloaded and need back-ups, ... Finally, it will be interesting to assess the fraction of team members' efforts on coordination tasks. The second objective is to predict the performance attained by the team when these coordination rules are in effect. Performance analysis using the System Effectiveness Analysis methodology, described in section 4.2.3, will be conducted on an example for which either centralized coordination or distributed coordination is used. The example to be used is a Navy example: the Combat Information

Center (CIC) of an Aegis ship. This example will be used throughout this task to illustrate the methodology of derivation and validation of coordination rules.

In order to perform the dynamic validation, there is a need to develop a new model of the interacting decision maker. The current modeling approach described in section 4.2.3 has some limitations that need to be addressed to approach the problem of distributed coordination in Adaptive Command and Control Teams:

- The current approach focuses on interaction between processes. There is a need to model in more detail the processes taking place, that is to open the "black boxes".
- Another type of problem is related to the stringent rule of firing of transitions of Petri Nets. For example, let us make a timeline analysis for the net displayed on Figure 4.2.2. If we assume that each process, represented by a transition, lasts 10 seconds, the derived timeline chart is displayed on Figure 4.2.7. One can see that because of the firing rules of Petri Nets, DM_4 does not start to perform his information fusion stage before he has received every piece of information: revised assessments of DM_1 and DM_2, and selected response of DM_5. This last piece of information is only provided at time 90, while the other pieces of information are received much earlier at time 40. There are therefore 50 seconds during which DM_4 is idle. One could easily imagine that this is not the case in the real world: DM_4 could start performing its information fusion and prepare its response according to the command from DM_3, received at time 70. The response could then be finalized, once DM_4 becomes aware of the response selected by DM_5.

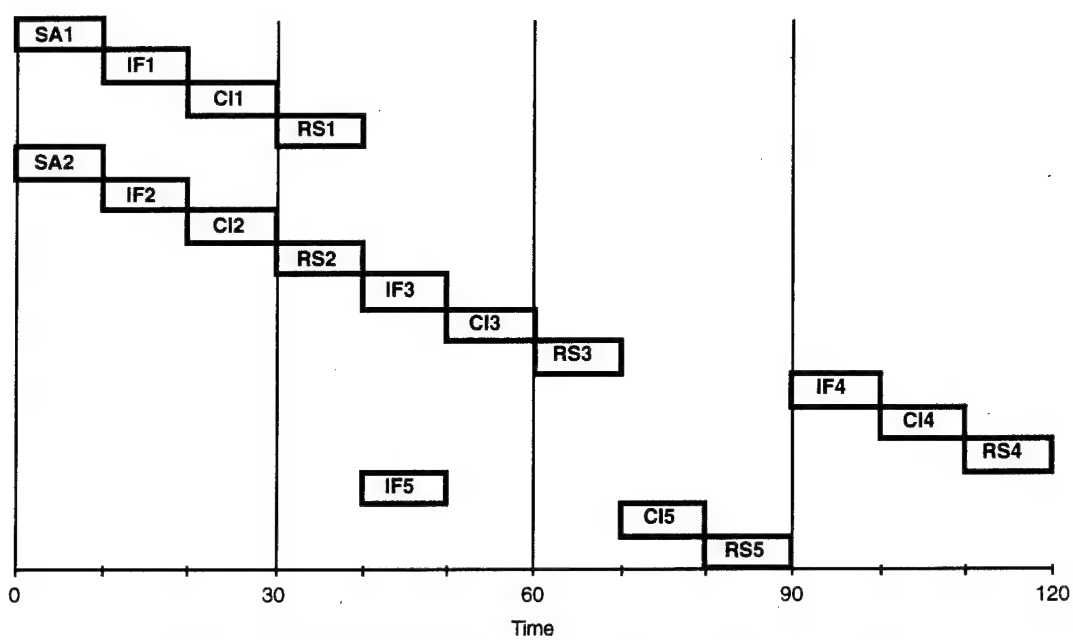

Figure 4.2.7 Timeline Analysis for the Model of Figure 4.2.2

- The organizations deal with only one input, which is supposed to arrive at the same time to every decision maker that can sense the environment. In the real world, even if correlated, inputs are provided at different times to the decision makers.
- Finally, the decision maker is considered to be memoryless and the modeling approach does not account for feedbacks. Remy's approach forbids the presence of loops and the decision makers can not "backtrack" in the light of external events or new information.

The revised model of the Interacting Decision Maker will be based on Colored Petri Nets so that it is compatible with the current approach to design and evaluate decision making organizations. The decision maker model will be an "intelligent" one, that is will be modeled as a knowledge rule-based system: the rule base will contain the task knowledge of the team member, the decision rules and the coordination rules, the working memory will contain deduced facts and data provided by the environment or other team members.

The model will account for uncertainty, asynchronicity of inputs, feedbacks and belief revision as follow: Instead of having rules of the type:

if E1 and E2 then H1

the model will use rules of the type:

R1: if E1 and H1(bel1) then H1 (bel2) with strength s1

where bel2 = f(bel1,s1)

R2: if E2 and H1(bel1) then H1 (bel2) with strength s2

where bel2 = f(bel1,s2)

f is a function to be defined that has to account for, if necessary, order bias or the discounting of information provided by other team members.

Assuming that E1 and E2 are provided at different times ($t_{E1} < t_{E2}$) and the initial belief (at time 0) toward H1 is B(H1) = bel0:

at time t_{E1}, B(H1) = bel1 = f(bel0,s1)

at time t_{E2}, B(H1) = bel2 = f(bel1,s2) = f(f(bel0,s1),s2)

There is an order bias if: $f(f(bel0,s1),s2) \neq f(f(bel0,s2),s1)$

4.2.5. Conclusion

The expected results of this task are that distributed coordination is possible and more effective than a centralized coordination, but there is a need to implement "circuit breakers" to avoid divergences and unstable behaviors. The major contribution of this research is that it addresses the issue of coordination for team reconfiguration that are not addressed in current systems design methodology. The proposed approach includes a procedure to generate automatically

coordination rules from a Colored Petri Net representation of the system, based on Colored Petri Nets invariants. Another contribution is the development of an "Intelligent" model of the decision maker that can be used to study a variety of other problems such as the effect of the limited human processing and memory capabilities or the impact of expertise.

4.3 INFLUENCE DIAGRAM REPRESENTATION OF DYNAMIC, DISTRIBUTED DECISION MAKING (Buede and Wagenhals)

This section describes the approach that GMU will take to accomplish the objectives of Task 5: Graphical Representation of C^2 Decision Making and Supporting Inference. The objective of this task is to use the rapidly developing field of influence diagrams and Bayesian networks to build graphical representations of decision making and supporting inference (intelligence) processing for command and control organizations. In particular, the emphasis is on distributed command and control organizations that are involved in evolving tactical situations and are likely to reorganize adaptively so as to remain effective. The focus for this task will be on the representation of decision making and inference processes within command and control.

This research is motivated by the large body of research in C^3 architectures dedicated to the generation of feasible distributed decision making organizations based on the formalism of Petri Nets. The mathematical expressiveness of Petri Nets along with the ability to execute them in simulation has facilitated the study of behavior and performance evaluation as the basis for selecting alternative architectures. While Petri Nets provide powerful techniques for analyzing the structure and the dynamics of the distributed decision making organizations, they rely upon an external representation of the decision and inference structure needed to make the C^2 process work. The decision theoretic community of researchers has been expanding formal approaches to modeling decision making processes under uncertainty and making inferences on the basis of incomplete and conflicting data. This research is focused on evolving the decision analysis literature to make it more amenable for incorporation into the Petri Net models.

The representation issues that we will address relate to distributed, concurrent, asynchronous, and interconnected command and control elements; the evolution of decision making tasks throughout a typical tactical mission; the exchange of information among elements associated with the evolution of time within a specific mission phase; and the impact of overarching environmental and threat uncertainties that inhibit the effective partitioning of the command and control elements. This activity will lead to new representation ideas in influence diagrams and Bayesian networks, enabling later research that addresses partitioning algorithms of decision making and inference tasks for the purpose of effective *allocation* of such tasks to command and control elements.

The next section describes the dynamic, distributed C^2 decision making environment as multi-level planning organizations that develop plans, disseminate directives, and monitor progress for revising or changing the plans. The second section defines the basics of influence diagrams, examines previous work that is potentially relevant to dynamic, distributed C2 decision making, provides the basics of our approach, and highlights issues relating to generalization and conversion to Petri Net models of the decision making process. The final section presents our plan for completing this effort.

4.3.1 Overview of Dynamic, Distributed Command & Control Decision Making

Military C^2 systems can be characterized as complex and highly dynamic information systems comprised of decision makers that are organized formally in a hierarchy and whose tasks are distributed over space and time. These decision makers (1) perform many tasks concurrently, (2) receive inputs asynchronously, and (3) require coordination and synchronization in order to affect the proper sequencing of events. This coordination is accomplished via message passing. The formal organization is hierarchical, but not necessarily the same as that of the functional hierarchy. A generic model of a C^2 system is shown in Figure 4.3.1. The military system represented in the diagram is a dynamical system comprised of a set of targets, a set of resources such as those comprising a naval battle force (ships, submarines, planes, etc.), and a command and control system (enclosed in the dark bordered box) to orchestrate its operation to meet the military objective using rules, doctrine, tactics, techniques, and procedures that have formulated and refined over centuries of battles and wars. The military objective of most combat operations is to attack and destroy a set of targets in the area of operations, represented as the battle space in Figure 4.3.1. The target set is not static. Targets move in time and location and are vulnerable to attack during certain time intervals called *windows of opportunity*. The challenge to the C^2 system is to locate and identify targets, determine their window of opportunity, and to get this information to the proper tactical elements so that the correct resources can be brought to bear upon the targets during the time window. Given operational constraints, these resources can be made available during *windows of existence*. One can think of the process as determining when the *windows of opportunity* and the *windows of existence* overlap. Actions that lead to these overlaps are the alternative plans from which the decision makers must select to maximize their objectives. This sequence starts with sensor systems that make observations about the battle space. These observations are reported to command centers where they are used by decision makers to assess the situation.

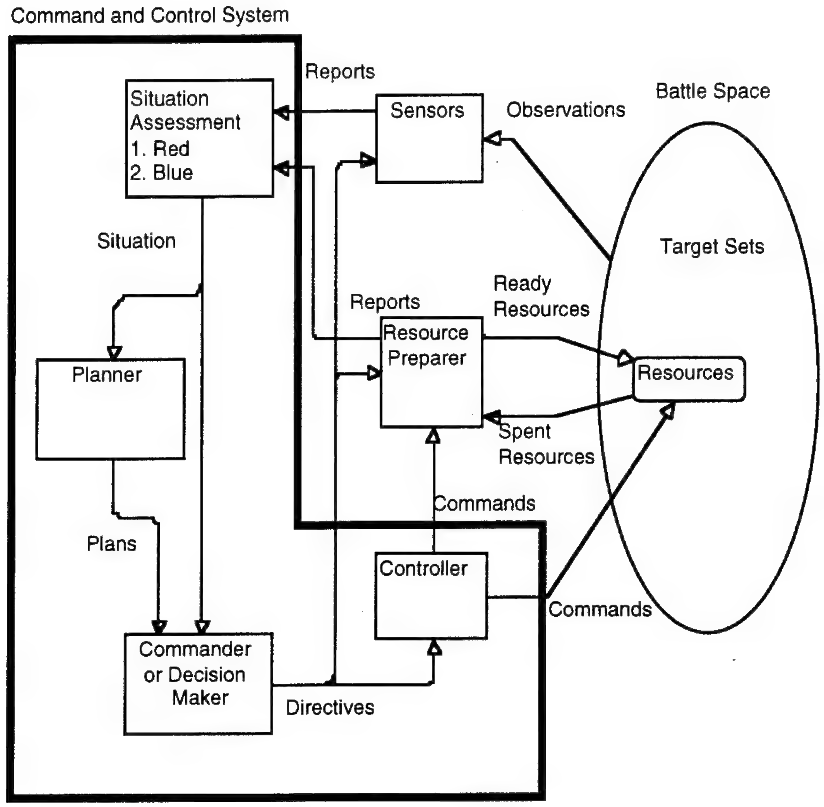

Figure 4.3.1 Generic Model of Command and Control System

The assessment of the situation for both adversarial (red) and friendly (blue) forces is based not only on the observation reports about the battle space, but also on reports from the resource maintainers on the status and availability of the resources at the disposal of the military system. Based on the estimates of the situation, planners devise plans that are comprised of actions that can be taken by the available resources. The Commander evaluates the alternate actions or sets of actions provided by the plans and selects the ones that the he believes will maximize the likelihood of meeting his objectives given the assessment of the current situation and the mix of resources required to implement each plan. These selected actions are passed to the resource preparers so that they know how to prepare the individual resources to be able to execute the actions in the plans. Directives are also provided to the execution controllers who implement the actions by selecting more detailed actions that are given to the resources during plan execution. Finally, directives are sent to those in charge of the sensor systems so that the priorities for information collection are known.

Most military C^2 systems are more complex then the one depicted in Figure 4.3.1. Just as the rules, doctrine, tactics, techniques, and procedures have been formulated and refined over

centuries, the structure and operational rules of the C^2 systems have also been standardized. Since the overall task to be performed is much too complex to be handled by a single decision maker, the need for multiple, distributed decision making structures is apparent. This requires a decomposition of the rules, doctrine, tactics, techniques and procedures; the allocation of the components of the decomposition (the tasks) to the distributed decision making nodes shown in the darkened border boxes; and the establishment of the rules of coordination between those nodes in order for the C^2 organization function to meet the overall objectives of the organization. Thus military systems are comprised of several systems that are operating in overlapping areas of the battle space requiring a hierarchical control structure as illustrated in Figure 4.3.2 where three hierarchical levels are shown. Level A is the highest level in this depiction. In such systems, higher level command entities receive situation reports from their lower level subordinate command entities. The higher level command entities also draw on a set of plans for the use of resources. Usually these plans are more general than the plans used by the lower level entities. In this figure, only two controlling elements are shown. In reality many such elements would exist. Figure 4.3.2 illustrates several important features of military C^2 systems. In real military systems, there are groups of specialized resource preparers that prepare their respective classes of resources. For example, in the naval context, there are separate preparers for aircraft, surface ships, and submarines. The resource preparers generally operate concurrently and independently making ready their class of resource with the proper load of weapons, fuel, etc. according to the current plan. This plan specifies when the resources are to be ready for a specified action in the battle space. While the resources may be prepared independently, in many cases they perform their actions in a coordinated fashion once in the battle space. This coordination is facilitated by the controllers. Figure 2 illustrates that there are multiple controllers. In order to accomplish the coordination of the activities of the resources, these controllers must in turn coordinate with each other (not shown in Figure 4.3.2) while they send specific commands or directions to the resources they are controlling. Figure 4.3.2 also indicates that there are many sensors making observations about the battle space. Different classes of sensors have different capabilities to detect, identify, locate, and determine the activity or state of objects and targets in the battle space. In addition, the higher level decision making organization may have access to sensor information that is not available to lower levels as shown by the sensor box at the top of the diagram. Such sensors may provide a wider view of the battle space than provided by sensors operated by the lower levels of the structure.

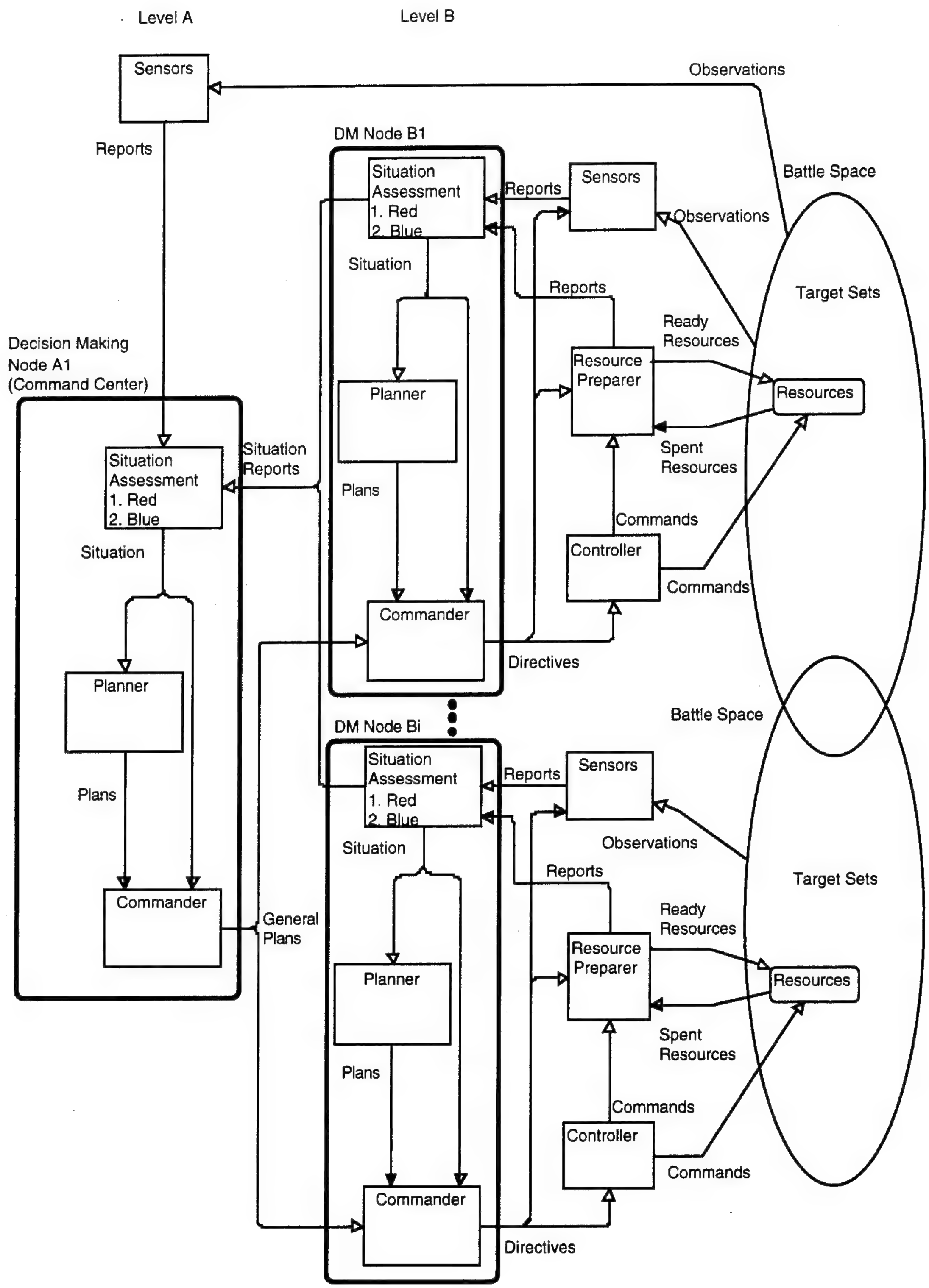

Figure 4.3.2 Multi-Level Command and Control Organizational Structure

As Figure 4.3.2 indicates, the activity responsible for assessing the situation does so in two arenas, the situation with regard to the adversary as well as the situation of the friendly resources. This situation is also divided into an estimate of the current situation and a projection of the future situation. Clearly the latter must be based on knowledge of the current plans that have been passed to the resource preparers and controllers. Figure 4.3.2 also indicates a partitioning of the planning responsibility. This partitioning is usually along the same lines as the partitioning of the resources preparers. Thus there are planners responsible for aircraft plans, surface ship plans, etc. The generation of these plans is based on the situation and occurs concurrently. This concurrent plan generation can result in potential conflicts between the plans, such as the need for too many resources to be refueled during a limited time window. These conflicts must be identified and adjustments selected to resolve the conflicts. In the architecture depicted in Figure 4.3.2, this process is the responsibility of the higher level from its overview vantage point. In other architectures, this conflict detection and resolution process may be distributed between command centers at the same level requiring different coordination paths and rules then those depicted in Figure 4.3.2.

From this first look at military C^2 systems, it is possible to derive several useful characteristics. Military C^2 systems can be characterized by both their structure and their dynamics. Structurally, they are composed of multiple layers of decision makers that work at increasing levels of generality, the higher one goes in the hierarchy. In general, the higher levels of the structure work on longer term, investment type decisions, that effect the situation in the battle space over a period of hours to days later. The lower levels work on shorter term, operational decisions having more immediate effects. One can view the decision making nodes as being specialized within each layer of the organization. For example, at one layer there may be a decision making node for logistics, situation assessment, and combat operations (strike, anti-submarine warfare, anti-surface warfare, and anti-air warfare in the naval context). Within each node of the distributed decision making organization, the decision problem consists of creating and selecting plans comprised of actions models that are based on the rules, doctrine, tactics, techniques, and procedures. These action models have action/time/space dimensions expressed in the form of a tuple (resource, action, location, time) plus some statements about initial constraints and the expected effect of the action. The action models vary in level of generality depending on the level within the organizational structure. For example, at a high level, an action model might be described in terms of moving a carrier task force to an offensive position 800 nautical miles from were they currently are in a period of one week. Subordinate levels will decompose and specialize the general plan into more detailed actions.

At a lower level the detailed action may specify that a particular cruiser should fire a cruise missile at a certain target location at a precise time.

There are several relevant dynamic characteristics of the C^2 organization. The first is that the organization is involved in repetitive decision making based on feedback. Each decision making node chooses a preferred trajectory (plan), observes the actual battle area trajectory, and then chooses corrective actions in terms of revised action models. Thus it is possible to characterize the decision making process as event driven rather than clock driven. With the event driven concept, decisions are made when it is believed that the actual trajectory deviates from the planned trajectory by more than a certain amount or when a new trajectory is directed by a higher level. Another dynamic characteristic is that the decision cycle times vary considerably with organizational level. This is because the higher levels are using more general plans with longer planning horizons than the lower levels.

Clearly, the decisions are being made under conditions of uncertainty since it is unlikely the system can know with certainty the outcome of any action or even the exact current situation. This uncertainty means that at any time there are many possible future states of the world, each with a certain probability of occurrence. Because of the uncertainty, most military C^2 systems develop contingency plans, each with its expected outcome given certain preconditions. These are available, and thus don't have to be generated, if the situation changes in ways that were not expected. These concepts mean that is unrealistic to model C^2 distributed decision making as a memoryless process. Clearly, the system retains the results of previous decisions, at least for a useful period of time.

A fourth characteristic of military C^2 systems is that they must continually deal with three types of variability. The first is variability in the inputs at each level, the second is the variability of the environment, and the third is the variability of the physical system that performs the distributed decision making functions. Examples of these variations include, inputs varying over time as the situation changes, environmental changes effecting the arrival rate of inputs within the system, and changes in the processing rates of systems within the organization to include failure of systems. To cope with this variability, modern C^2 systems must be adaptable. To cope with this variability, research in the C^3 Architecture design suggests three types of adaptation: change the physical systems and their connectivity, change the coordination schemes, or change the task allocation.

The enemy situation assessment is a very complex process due to the uncertainty and variability described above. This continuous process must accept reports from multiple sensor systems

that are received asynchronously. Because the observations are from sensors with a diversity of capability and accuracy, the process must attempt to resolve many ambiguities in the reporting process. Furthermore, it must resolve these reports into hypothesized entities to reduce the tendency to count the same entity multiple times and to enable inferences about entity locations, entity identifications, association with other entities, and entity activities at the time of the observations as well as in the future. Once a current estimate is established, it is necessary to project future estimates.

The friendly situation assessment is less complex than the enemy situation assessment. This is because more reliable, timely and accurate reports will be available to the assessor. Nevertheless, there is still some degree of uncertainty and ambiguity is still possible.

The plan generation process is partitioned among staff decision makers based on a specialization of the process for a given class of resource. Inputs include the situation (enemy and friendly, both current and projected), the current set of plans, and adjustment instructions from the higher level decision maker. In this concept, the planners actually modify existing plans. Certain changes in the situation trigger the plan modification process. In this concept, each decision making node creates new plans or plan modifications asynchronously and concurrently. The organizational design must address the coordination among the planners to avoid conflicts and to synchronize the process.

In order to bound the scope of the research, we propose to concentrate initially on three levels of hierarchical structure of the C^2 distributed decision making structure. We believe that this will simplify the formulation of the problem without loss of generality to hierarchies with more than three layers. The basic assumption is once the representation of the process has been defined for three layers, it can be expanded to additional layers by recursively applying the process to the middle level within the structure in order to add the additional layers. The middle layer will be the focus with the upper and lower layers serving as boundary layers, sending inputs to and receiving outputs from the middle layer.

Before moving on to the discussion of influence diagrams, it is useful to summarize the main characteristics and issues that have been identified from the analysis of the block diagrams.

1. We plan to represent only three levels of the hierarchical structure or the distributed C2 decision making system. The method developed should be easily extended to systems with more layers.

2. The representation of the situation assessment process, will require methods that deal with uncertainty in a manner consistent with decision making.

3. The main decision making process at each level involves the selection of alternative plans. The generation of the feasible alternatives is also a complex problem that will not be addressed in detail in this effort. We will represent the alternative plans as either a generalization/specialization or a composition/decomposition hierarchy. It will be assumed that the alternative generation process exists and creates a set of feasible alternative plans from which the decision maker selects the "best".

4. The decision making process is guided by an overarching set of values dedicated to winning the war. As the situation changes, the decision makers adjust their trade offs among present versus future losses and current losses versus battle gains.

4.3.2 Representing Dynamic, Distributed C2 Decision Making with Influence Diagrams

Overview of Influence Diagrams and Bayesian Networks

Influence diagrams are a graph theoretic representation of a decision. After significant research by Howard (1990), Shachter (1986, 1990) presented the requirements and algorithms needed to transform an influence diagram from solely a communication tool into a computation and analysis tool capable of replacing the standard decision analytic tree. Significant additional research continues into influence diagrams for structuring decision problems, defining the underlying mathematics and graph theory of influence diagrams, and analyzing decision problems. When properly implemented, decision trees and influence diagrams provide identical solutions to the same problem. They are referred to as isomorphic since the decision tree can be converted to an influence diagram, and vice versa.

An influence diagram may include four types of nodes (decision, chance, value, and deterministic), directed arcs between the nodes, a marginal or conditional probability distribution defined at each chance node, and a mathematical function associated with each decision, value and deterministic node. Each decision node, represented by a box, has a discrete number of states (or decision options) associated with it; chance nodes, represented by an oval, must be discrete random variables. Deterministic nodes are represented by a double oval. A value node may be represented by a rounded cornered box, diamond, hexagon, or octagon.

An arc between two nodes (shown by an arrow) identifies a dependency between the two nodes, see Figure 4.3.3. An arc between two chance nodes expresses relevance and indicates the need for a conditional probability distribution. An arc from a decision node into a chance or

deterministic node expresses influence and indicates probabilistic or functional (respectively) dependence. An arc from a chance node into a deterministic or value node expresses relevance, that is to say the function in either the deterministic or value node must include the variables on the other ends of the arcs. An arc from any node into a decision node indicates information availability; that is, the states of these nodes are known with certainty when the decision is to be made.

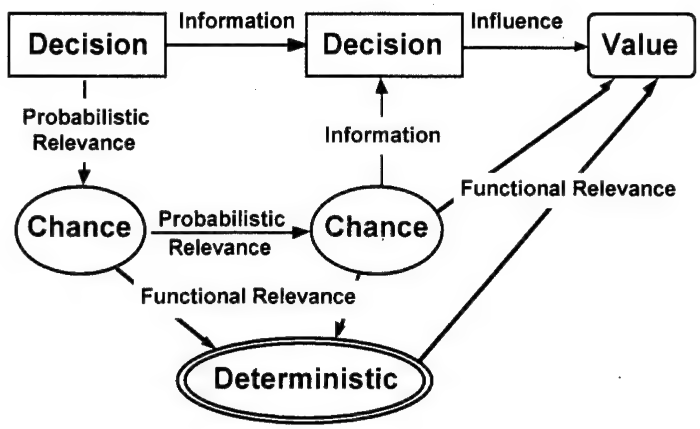

Figure 4.3.3 Node and Arc Types in an Influence Diagram

The decision node represents a logical maximum (minimum) operation, that is, choose the option with the maximum (minimum) expected value or utility (cost). A deterministic node can contain any relevant mathematical function of the variables associated with nodes having arcs into the deterministic node. A value node also can contain any mathematical function of the variables with arcs entering the value node. In addition, the mathematical function in the value node defines the risk preference of the decision maker.

A well-formed influence diagram meets the following conditions: (1) the influence diagram is an acyclic directed graph, that is, it is not possible to start at any node and travel in the direction of the arcs in such a way that one returns to the initial node; (2) each decision or chance node is defined in terms of mutually exclusive and collectively exhaustive states; (3) there is a joint probability distribution that is defined over the chance nodes in the diagram that is consistent with the probabilistic dependence defined by the arcs; (4) there is at least one directed path that begins at the originating or initial decision node, passes through all the other decision nodes, and ends at the value node; (5) there is a proper value function defined at the value node (that is, one that is defined over all the nodes with arcs into the value node); and (6) there are proper functions defined for each deterministic node. An influence diagram that is well formed can be

evaluated analytically to determine the optimal decision strategy implied by the structural, functional and numerical definition of the influence diagram.

Decision analysts and Bayesian probabilists have developed another graph-theoretic construct, called Bayesian or belief networks, that is critical to the modeling of C^2 decision making processes. A Bayesian network is (1) a directed graph representing a factorization of a joint probability distribution over n random variables, (2) the joint probability distribution of the n random variables, and (3) a computational architecture for updating the joint probability distribution via Bayes rule as new information is received about any of the random variables. The directed graph contains nodes (one for each random variable) and arcs. The arcs specify which conditional distributions have been chosen in the representation of the joint distribution. For example, the first Bayesian network in Figure 4.3.4 has a joint distribution over x_1, x_2, and x_3 that is represented by $p(x_3|x_2)$, $p(x_2|x_1)$, and $p(x_1)$. The absence of an arc from x_1 to x_3 is as important as which arcs are present; the absence of an arc from x_1 to x_3 means that x_3 is conditionally independent of x_1, given the value of x_2. Note, that no joint probability distribution can be represented by a directed graph with a cycle (a directed path that returns to its origin). For a three variable problem there are six possible representations of the joint distribution for the case of full probabilistic dependence; one of which is shown on the right of Figure 4.3.4.

$p(x_1, x_2, x_3) = p(x_3|x_2)\, p(x_2|x_1)\, p(x_1)$ $\qquad$ $p(x_1, x_2, x_3) = p(x_3|x_2, x_1)\, p(x_2|x_1)\, p(x_1)$

Figure 4.3.4 Two Representations of Three-Variable Bayes Nets

In addition to representing a joint probability distribution, a Bayes network provides an inference engine for updating the uncertainty of the joint distribution given new information such as sensor reports. A significant number of probabilistic propagation algorithms have been devised (Pearl, 1988; Neapolitan, 1990) for networks containing only discrete random variables; the case of normally distributed continuous variables and mixed discrete/continuous variables have also been addressed (Chang, 1995) in the literature. The algorithms accept sensor reports, in the form of likelihoods, at any node in the graph as well as changes to the prior distributions at any root node. It is not germane to this discussion to review these algorithms, other than to mention that they (1) can be categorized as exact and simulation-based

approximations, (2) are very efficient because they take advantage of the conditional independence present in the graph and (3) are computationally parallelizable. The exact algorithms may be NP-hard for large, highly connected graphs, but this is an initialization issue only.

For multiple sensor and multiple platform applications of data fusion, the sensors may provide inputs to one or more of these nodes. So the Bayesian network is an integration mechanism for the inputs of the sensors; this is the function performed during situation assessment.

An important element of our approach to modeling dynamic, distributed C^2 decision making is to combine Bayesian networks and influence diagrams. Figure 4.3.5 shows an influence diagram for a simple operational scenario with one target in the area for a fighter or attack aircraft. There is one fundamental objective - mission success - with three specific objectives: mission target destroyed, fratricide, and survival. At any point in time the trade-offs amongst objectives can be modified by a higher level C^2 organization, changing the best course of action. The pilot's action at this point in time is to shoot at the target, wait longer or begin evasive maneuvers. There is uncertainty about the success of shooting or evading depending upon the target's status and activity, which includes a number of variables that will appear in the Bayesian network. Finally, there is uncertainty about whether the target will shoot at the pilot's aircraft ("own ship") first, making the pilot's survival a question.

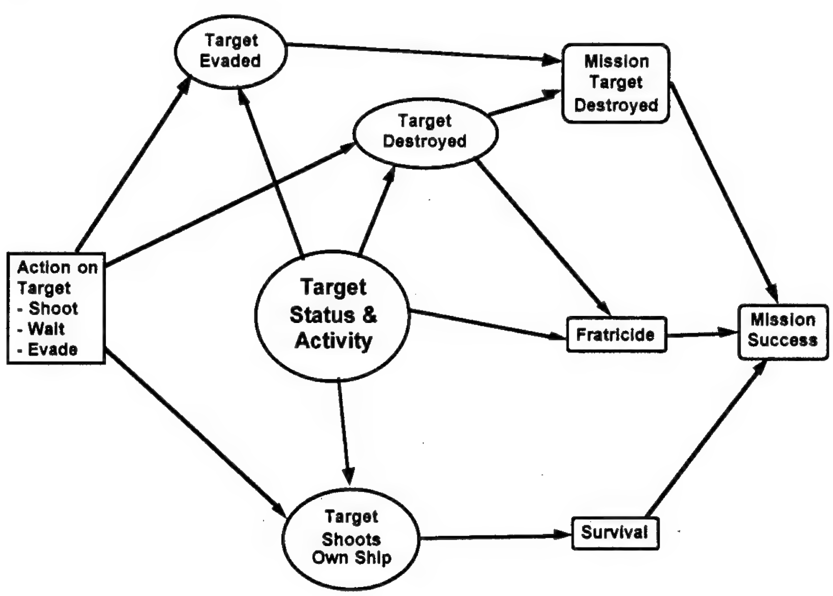

Figure 4.3.5 Influence Diagram for a One Target Scenario

Figure 4.3.6 shows the Target's Status and Activity node in Figure 4.3.5 exploded into the Bayesian network that is maintaining status of the uncertainty about the target's type (e.g., F-15 vs. MiG-29), class (fighter vs. bomber), nature (friend vs. foe), range, aspect angle, and activity. This figure shows a radar report (eight sided node) has been used to update the target's range, and track data (eight sided node) has been used to update our uncertainty about the target's aspect angle and target class. For example, the target's velocity estimate might serve as an estimate of the aspect angle; the target's altitude and speed may provide valuable information about whether the target is a fighter or bomber.

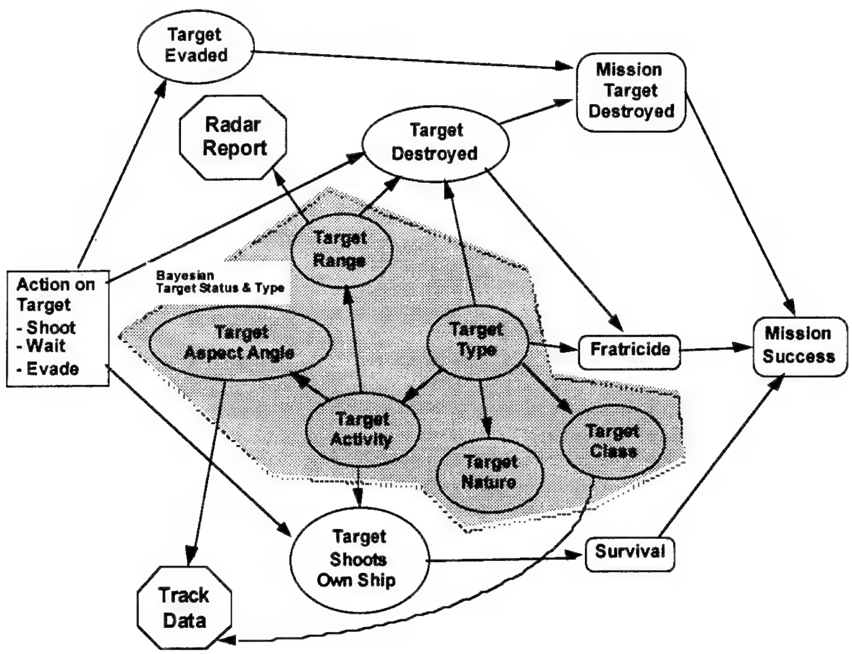

Figure 4.3.6 Influence Diagram with Embedded Bayesian Network

There are several important features to notice about the Bayesian network. It is common practice in Bayesian networks for the key uncertainties, called target nodes, to be source or border nodes, that is nodes that have no parents. These nodes are initialized with a prior distribution that is then updated as evidence is received, hence the name Bayesian network. In the above Bayesian network within the influence diagram one node satisfies this characteristic: target type. Other nodes are added to the Bayesian network for which it is often easier to collect information; target activity, target nature (friend versus foe), and target class are children of target type in Figure 4.3.6, target range and target aspect angle are children of target activity. These nodes are children of the target nodes and contain probability distributions conditioned on the values of the target nodes and occasionally each other. Since the target nodes are embedded in the influence diagram, the target nodes will usually be on one or more directed

paths from the decision nodes and ending at the value node(s). In Figure 4.3.6 none of the chance nodes in Bayesian network are on these directed paths. The non-target nodes in the embedded Bayesian network are on paths directed away from the target nodes and therefore are unlikely to be directly relevant to the decision. Shachter (1990) showed that relevancy is determined by whether a chance node is in the Markov blanket of the nodes on the directed paths from the decision nodes to the value node(s), called the decision paths. The Markov blanket of the nodes on these decision paths is defined to be set of nodes that satisfy one of the following criteria: parent of one or more nodes on the decision paths, child of one or more nodes on the decision paths, and a parent of a child of one or more nodes on the decision paths. Target type is in the Markov blanket because it is the parent of two nodes on the decision paths, target destroyed and fratricide. Target activity and target range are also members of the Markov blanket of decision paths. Nodes that are not members of the decision paths or the Markov blanket of the nodes on the decision paths can be immediately pruned (thrown away) when it is time to solve the influence diagram. Therefore the only effect that nodes such as target aspect angle and target class have is as a conduit for evidence that updates the nodes in the Markov blanket or on the decision paths as sensor reports are received.

Previous Work on Dynamic and Distributed Decision Making

Tatman and Shachter (1990) define a generic influence diagram structure (Figure 4.3.7) for a multi-period decision process (e.g., dynamic programming). Important elements of this multi-period decision process structure are (1) the segmentation of the decision process into distinct stages in which there are one or more decision nodes, one or more chance nodes and a single value node fed by the decision and chance nodes in that stage only, (2) the informational edges from the chance nodes of that stage for each set of decision nodes in that stage, (3) probabilistic relevance nodes that cross from one stage to the next stage's chance nodes but no farther, and (4) a final value node that aggregates the value nodes of each stage.

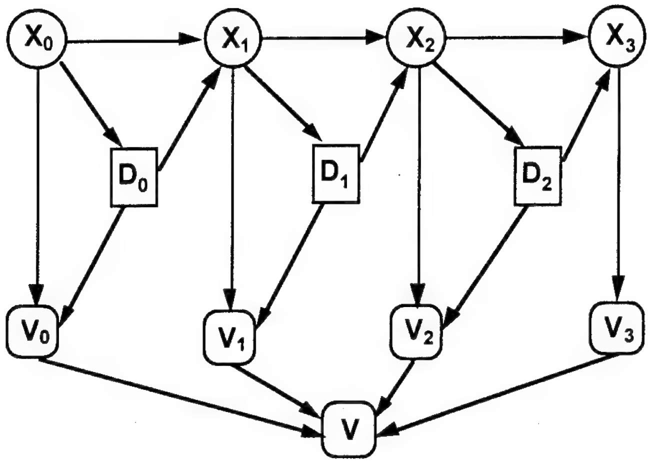

Figure 4.3.7 Generic Three Stage Decision Process

Pete, Pattipati and Kleinman (1995 and forthcoming) have presented two approaches for designing organizational structures using the concepts of influence diagrams. Both approaches involve using influence diagrams to model the decision making structure. (Note they choose to drop the concept of value node.) In both cases they constrain the organization design to meeting the assumption that the decision is equivalent to the optimal decision that a centralized decision maker would obtain; they define an organization meeting this constraint as a congruent organization. In both cases they set up an optimization problem of finding the minimum communication network necessary to support the congruent organization in reaching its optimal decision.

One approach (Pete et al., 1995) addresses distributed decision making organizations in a non-dynamic environment that are obtaining situation assessment reports about the environment but are not able to affect this environment based upon their decision options. Such organizations are likely to be information processing organizations, such as an intelligence element. While this work is interesting, it completely ignores the entire literature on Bayesian networks which provide a powerful and consistent approach for examining the situation assessment problem.

The second approach of Pete et al. [forthcoming] addresses distributed, dynamic decision making organizations. Each element of the organization receives imperfect reports (Y in Figure 4.3.8) at time t about the environment (H) allocated to it and then makes a decision (D) that impacts the environment allocated to it in time t+1. It is also assumed that the states of each segment of the environment at time t impact the state of every segment of the environment at time t+1. It is also assumed that at any time t, the decision makers make their decisions independently, but then communicate their decisions to all other decision makers so that this

information is available at time t+1. The cost (C) that is being minimized is a function of H and D at any point in time for each decision maker. Total costs can be aggregated for each decision maker and then across decision makers. Note that Pete et al drop these value or cost nodes. Figure 4.3.8 shows these assumptions in an influence diagram similar to the one developed by Tatman and Shachter (1990). The results developed by Pete et al. are quite valuable but limited to a single decision making layer, not the hierarchical structure developed above that is typical of C^2 organizations. Also, the situation assessment model representing environmental uncertainty is more neatly structured than is common in hierarchical C^2 organizations. Finally, given the voluminous communications typical of C^2 organizations it is unlikely that such an organization would be interested in minimizing a very small subset of those communications in determining its organizational structure.

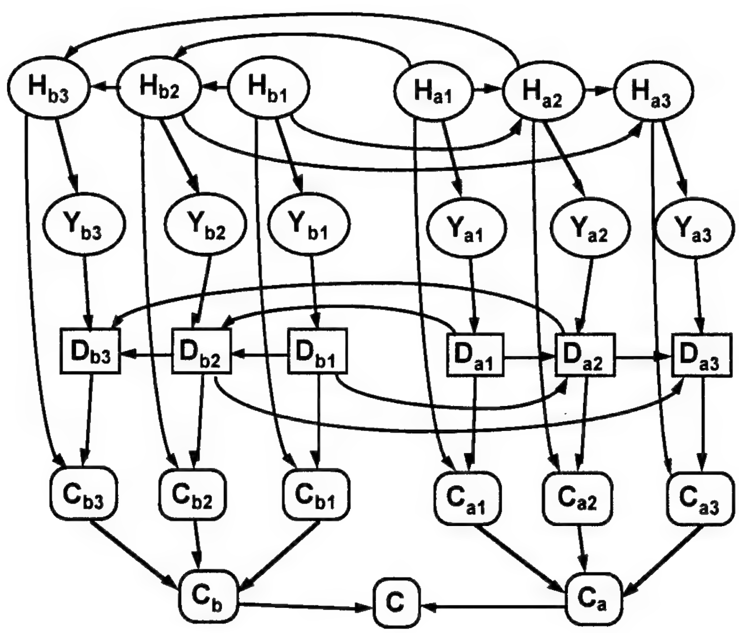

Figure 4.3.8 Multiple Decision Maker Network of Pete et al [forthcoming]

Dynamic Decision Networks -- Our Approach

Our approach is to use dynamic decision networks (DDN), a set of interconnected influence diagrams and Bayesian networks. Influence diagrams are a decision analytic and graphical construct for representing a decision problem in terms of decision, uncertainties and values, and the probabilistic and informational interactions among them. Bayesian networks are inference engines that model complex interdependent stochastic processes.

There will be a DDN for each organizational element of the C^2 organization. We will begin by modeling the decision process of each organizational element of the C^2 organization with a single decision node that addresses the options available to the specific organizational element. For each influence diagram there will be one or more Bayesian networks that maintain the current uncertainty on all relevant random variables, based upon sensor and intelligence reports. These chance nodes represent the situation assessment being done by that element. Since the situations for each organizational element are linked to the same battlefield events, the Bayesian networks across organizational elements are likely to be part of one large connected graph.

We will model the distributed decision making process as a set of event-driven timed processes that are based on influence diagrams with imbedded Bayesian networks. Each timed process will be triggered by an initiating event that causes a specific organizational element to face a decision to change its current activity or to continue with the present activity. These triggering events occur according to the underlying processes occurring in the battlespace. These events are observed by blue sensors and sources, which report the occurrence and description of the events to the decision making process. We define the time difference between triggering events for a specific decision making entity to be a time slice. When the battle with the enemy is hot (e.g., weapons are being fired) the time difference between triggering events (time slice) may be on the order of minutes for the high level C^2 organizations that we are considering. When there is no direct contact between forces, a time slice might be hours long.

After a triggering event occurs initiating a time slice, we will assume there is directed graph that links the decision nodes in the DDNs of each functional organization, creating a single DDN for the group of C^2 organizations (Figure 4.3.9). This assumption means that we have determined that the functional organizations will make their decisions in a pre-defined order. This assumption would be very problematic at lower levels of the C2 process and for operational units that must decide to return fire in seconds or less and cannot wait to find out what other sibling organizations are doing. However, at higher levels of C^2 organizations we can assume that the decision order amongst C^2 elements is directly related to their primacy in facing the current situation, as determined by the commander of the organization being modeled. As the situation changes, this commanding officer can issue directives that change the primacy of subordinate C^2 elements, thus changing the order of the decision nodes in the overall DDN.

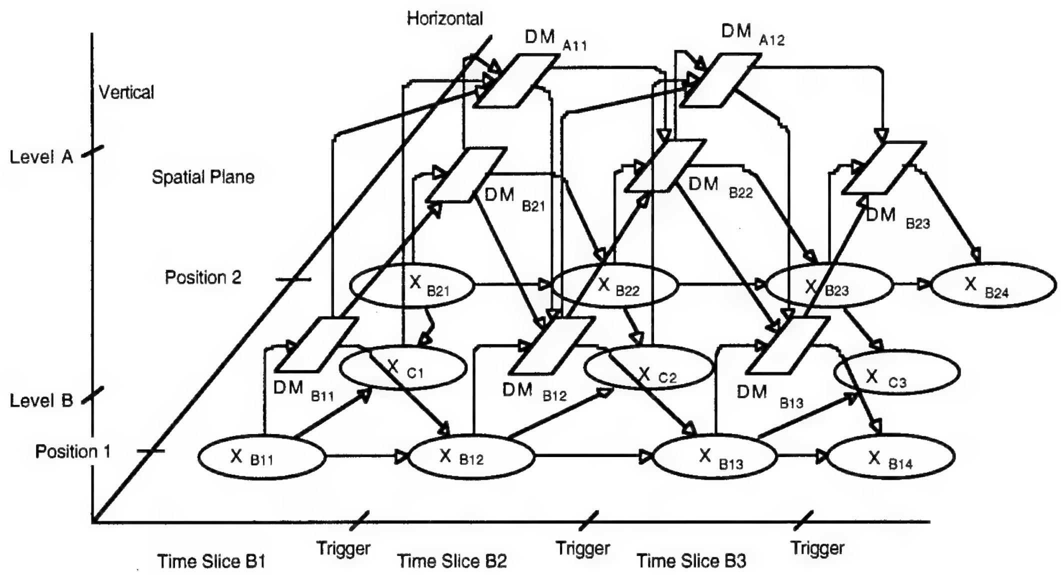

Figure 4.3.9 Spatial/Temporal Depiction of the Distributed Decision Making Process

Figure 4.3.9 is a notional depiction of the above concepts. It illustrates in a Spatial/Temporal space, the dynamic distributed C^2 decision making process representation of the type of organization structure that was discussed in Figure 4.3.2. The spatial plane depicts the hierarchical and geographical dispersion of the distributed decision making process. Although only two hierarchical decision making levels are shown in the diagram for clarity, three levels will be addressed in this research as discussed previously. The axis labeled "horizontal" represents the geographically separated decision making units at the same organizational level. Two units are shown at Level B and their single supervisory decision making unit is shown at Level A. The real-time battle area activities of the resources, sensors, and controllers are represented as random variables in ovals. The Time axis represents the evolution of time and indicates both the time when triggering events occur and the subsequent time slices.

The graph placed in the Spatial/Temporal space is a pseudo influence diagram. Each of the vertices in the graph has an index that indicates its position in the spatial/temporal space. For example, the index B11 indicates Level B, Position 1, Time Slice 1. Two types of vertices are shown, boxes and ellipses. The boxes represent the decision making units. Within each box exists an instance of the generic decision making node capable of situation assessment and plan selection. The ellipses represent the chance nodes in the influence diagram. Shown at the lowest level, they represent the situation in the battle space for each time slice. There are multiple chance nodes at this level representing the portion of the battle space that each decision making unit can influence or "see". For example, the decision making nodes in position 1 at

level B can only influence (and be influence by) the portion of the battle place with index C_{1i} The decision making unit at Level A can "see" a composite situation on the battle field represented in the chance nodes with the single index (X_{C1}, X_{C2}, X_{C3}, etc.). The values of these nodes are conditioned by the value of the nodes influenced by the decision makers at Level B. The value nodes of the influence diagrams and their connecting arcs have been omitted from the diagram for clarity purposes.

In operation, the influence diagram of Figure 4.3.9 would execute sequentially, stepping from state to state as each triggering event occurs. The decision making node at Level A would observe the situation and would have "wait" as one of its options. Thus, it would not send any update to the lower level unless and until the situation warranted. Recall also that the decisions send by Level A are generalizations or aggregations of the decision making alternatives of the Level B. Consequently, they have a longer planning horizon then Level B. Thus, Level A could wait for several time slices before observing sufficient change in the situation that was a result of the influence by the decisions made at Level B.

The pre-defined ordering of the decision making process is depicted by the arcs that interconnect the decision making nodes. As indicated by the direction of the arcs in Figure 4.3.9, the decision maker in position 1 at level B would make its decision before its sibling decision making node at position 2. The diagram depicts that this ordering continues over the three time slices. If the situation changed, so that decision making node DM_{B2} should have primacy, the supervisory decision making node at Level A could issue instructions that would change the direction of the arcs between the decision making nodes at Level B, thus adapting the organizational structure to match the changed situation.

Note that the decision making node at the higher level has several ways of adapting the organization to changes in the situation. These methods include changing the generalized plans that are issued to the lower levels, changing the ordering of the decision making, and adjusting the variable values in the value nodes of subordinate decision making nodes. The latter, in effect, chances the risk preference of the subordinate decision maker.

One of the technical challenges of this task will be the representation of the distributed situation assessment process for the interconnected decision making nodes. The process of conducting situation assessment for either friendly or enemy forces is complicated by the uncertainty surrounding reports of humans and sensors, the vast amount of information that must be examined to understand where all of the pieces are and what that means at a higher level of abstraction for the force, and the interconnection of information sources. Figure 4.3.10 is an

illustrative Bayesian networks for situational assessment for the commander who has wrfare commanders for AAW, ASW, and ASuW reporting to him. Figure 4.3.11 is an illustrative Bayesian network for AAW situation assessment for the AAW commander. Note that in both cases there are nodes that capture the current situation and others that estimate the future situation based upon the current situation. In reality there might be several future time periods in question at each level of decision making; but the time intervals might be longer for the higher level commander than for the lower level commanders. Figure 4.3.12 illustrates the common nodes for the assessment of the current situation for the two levels depicted in Figures 4.3.10 and 4.3.11. It is likely that the estimates of the current situation for nodes that are shaded darkly in Figure 4.3.12 for the higher level commander might be received directly from the lower level commanders. Similarly, the estimates of the nodes that are shaded lightly in Figure 4.3.12 are likely to be sent from the higher level commander to the lower level commanders. Note that not all of these nodes (e.g., current enemy tactical objective) need be used by the lower level commanders. Also there are a number of nodes in the lower (higher) level situation that need not be seen by the higher (lower) level commanders.

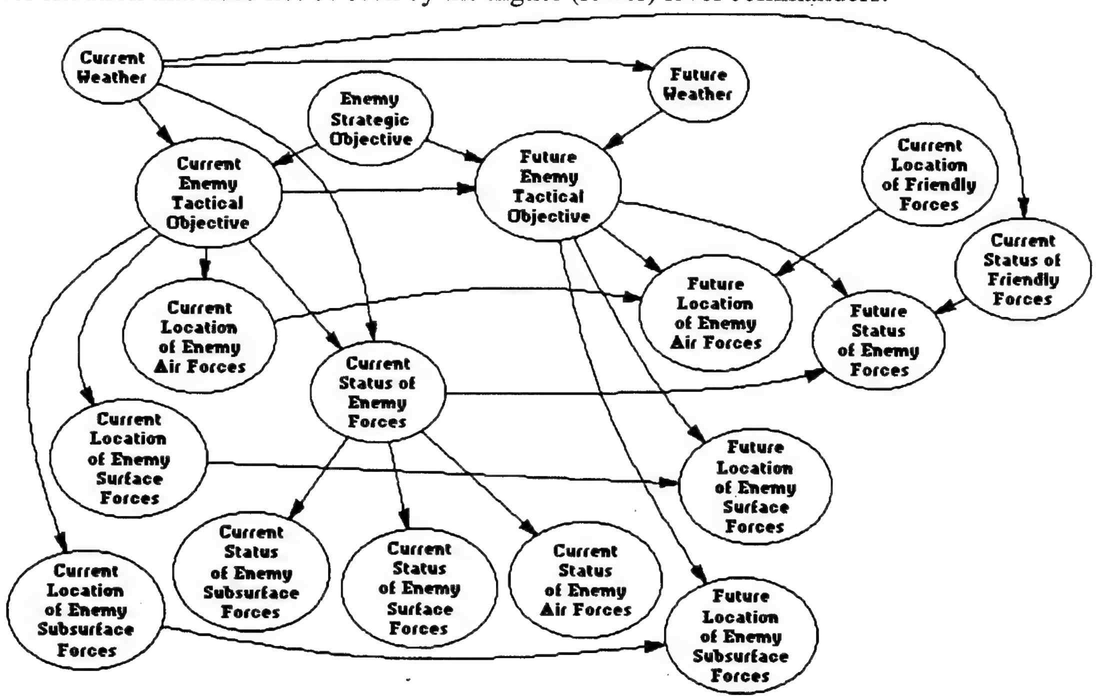

Figure 4.3.10 Prototypical Bayesian Network for Overall Situation Assessment

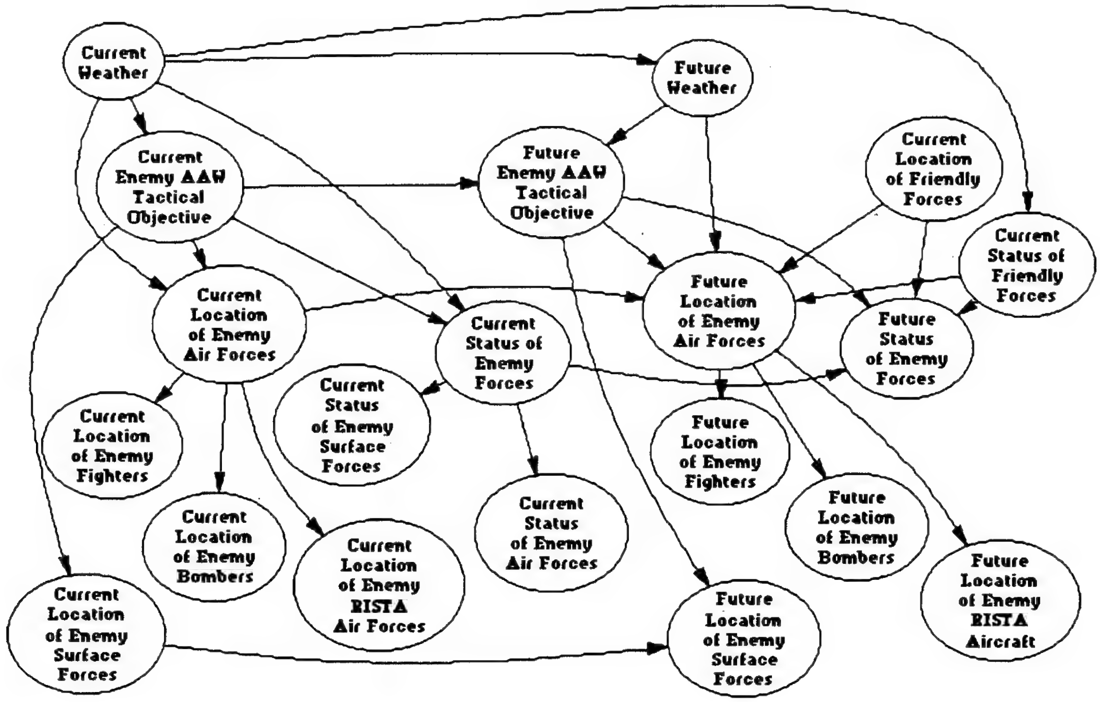

Figure 4.3.11 Prototypical Bayesian Network for AAW Situation Assessment

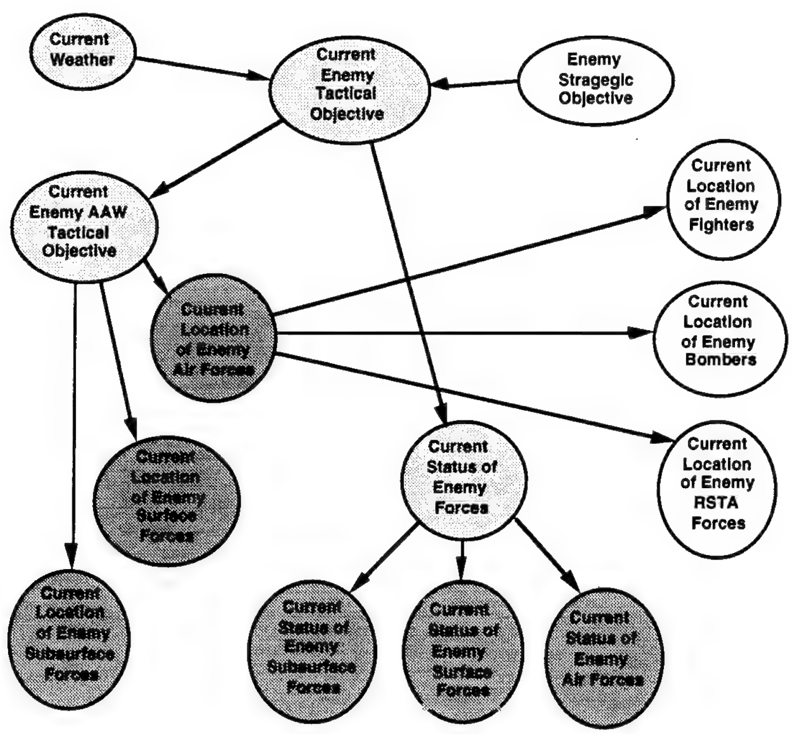

Figure 4.3.12 Common Nodes for Current Situation of Figures 4.3.10 and 4.3.11

The names of the nodes and their underlying states are not nearly so important as the general topological characteristics of the networks; by topological characteristics we mean the connectivity amongst the nodes of the networks. In each case the networks are significantly multiply-connected. A multiply-connected network is one in which it is possible to find more than one semipath between two nodes; a semipath is connected set of links for which we can ignore the direction of the arcs. In addition, a number of the nodes in the two networks have identical names, meaning the processing should be centralized or the distributed organizational elements must be relying on each other for frequent and highly specific updates. In adapting from one organizational design to another, the centralized Bayesian network should be used to determine which nodes to allocate to each element. The nodes cannot be partitioned amongst the organizational elements because that would too severely restrict the needed information for decision making. At the same time only one organizational element can be given primary responsibility for each node. In addition, we will not be able to exploit Heckerman's (1990) ideas of similarity networks which have developed for problems with a single target node or source. As can be seen below both of these networks have multiple sources.

The issue of maintaining a coherent tactical picture (situation assessment) amongst the distributed, multi-level decision makers is not easy in a rapidly changing environment. Coherency means that they all see their own needed subset of the "centralized" view of the situation during the same time slice. Making this happen is problematic because the interconnected Bayes Net is designed to capture the situation from every perspective (coherence) but the delays in the communications system passing the updates can create problems in coherence. Solving this problem correctly requires that we move to an event driven representation of time. Representing and testing these phenomenon not only requires the Bayes Net representation, but may also require a dynamical model such as the Petri Net, particularly to capture the consistency phenomenon.

Conversion issues

One of our ultimate objectives for this project is to be able to convert the representation of the distributed decision making process of a C^2 organization as a collection of separate influence diagrams into the Petri Net representation that has been used to great advantage in the research on the design of C^3 architectures. Petri Nets in general, and hierarchical colored Petri Nets in particular, have been extremely useful in creating executable models of C^3 systems that can be used in simulation to uncover behavior characteristics and to explore performance issues where temporal aspects inducted by delay in communications paths are important. To refine our description of this aspect of the research problem we need to shift from the block diagram

representation technique of Figures 4.3.1 and 4.3.2 and the graphical representation of the influence diagram of Figures 4.3.5 and 4.3.6 to that of a Colored Petri Net (CPN) of the intelligent command and control node (Levis, 1992).

4.3.3 Plan for Remaining Work

Our activities on this task for the rest of this contract are:

a. continue to refine the descriptive model of military dynamic, distributed C2 elements,

b. further define the topological properties of Bayesian networks for situation assessment,

c. assess the Bayesian network topological implications on the Markov Blanket of the influence diagram for the C^2 organization,

d. examine the possibilities for representing the distributed decision making process of C^2 organizations as a collection of separate influence diagrams,

e. further develop an event-driven representation of the time dynamics that fits the notions of influence diagrams and Bayesian networks as well as the five stage Petri Net model (Levis, 1992) of the C^2 decision making process.

4.4. SUMMARY

Progress achieved on all tasks during the six months period has been reported.

4.5 REFERENCES

Andreadakis, S. K., and A. H. Levis (1988). Synthesis of Distributed Command and Control for the Outer Air Battle. *Proc. 1988 Symposium on Command and Control Research*, Science Applications International Corporation, McLean, VA. pp. 352-364.

Athans, M. (1982). "The Expert Team of Experts Approach to Command and Control (C2) Organizations." *IEEE Control Systems Magazine*, Vol. 2, No. 3, pp. 30-38.

Boettcher, K. L., and A. H. Levis (1982). Modeling the Interacting Decisionmaker with Bounded Rationality. *IEEE Trans. Syst., Man and Cybern.*, Vol. SMC-12, pp. 334-344.

Cannon-Bowers, J. A., E. Salas, and S. A. Converse (1991). Cognitive Psychology and Team Training: Training Shared Mental Models of Complex Systems. *Human Factors Society Bulletin*, Vol. 33 No. 12, pp. 1-4.

Chang, K.C. (1995). Symbolic Probabilistic Inference with Both Discrete and Continuous Variables. *IEEE Trans. Syst., Man and Cybern.*, Vol. 25, No. 6, pp. 910-916.

Davis, L. (1991). *Handbook of Genetic Algorithms*. Van Nostrand Reinhold, NY.

Demaël, J. (1989). On the Generation of Variable Structure Distributed Architectures. Report LIDS-TH-1869, M.S. Thesis, Laboratory for Information and Decision Systems, MIT, Cambridge, MA.

Demaël, J. J. and A. H. Levis (1994). On Generating Variable Structure Architectures for Distributed Intelligence Systems. *Information and Decision Technologies,* pp. 233-255.

Design/CPN: A Reference Manual, (1991). Meta Software Corp., Cambridge, MA.

Duffy, L. (1993). Team Decision Making Biases: An Information Processing Perspective. In G. A. Klein, J. Orasanu, R. Calderwood, and C. E. Zsambok (Eds.) *Decision Making in Action: Models and Methods*. Ablex Publishing Corp., Norwood, NJ

Genrich, H. (1987). Predicate Transition Nets. In *Advanced Course on Petri Nets 1986*, Lecture Notes in Computer Science, Springer-Verlag, Berlin, Germany.

Heckerman, D. E. (1990). Probabilistic Similarity Networks, Ph.D. dissertation, MIS Department, Stanford University, Stanford, Ca.

Holland, John H. (1975). *Adaptation in Natural and Artificial Systems*. The University of Michigan Press, Ann Arbor, MI.

Howard, R. A. (1990). From Influence to Relevance to Knowledge. In Oliver, R. and J. Smith (Eds.), *Influence Diagrams, Belief Nets and Decision Analysis*. John Wiley & Sons, NY.

Jensen, K. (1987). Colored Petri Nets. In *Advanced Courses on Petri Nets 1986*, Lecture Notes in Computer Science, Springer-Verlag, Berlin, Germany.

Kleinman, D. and D. Serfaty (1989). Team Performance Assessment in Distributed Decision Making. In R. Gilson, J. P. Kincaid and B. Goldiez (Eds.), *Proceedings of the Interactive Networked Simulation for training Conference*. Naval Training Systems Center, Orlando, FL.

Levis, A. H. (1992). A Colored Petri Net Model of Command and Control Nodes. In C. R. Jones (Ed.) *Command, Control and Communications: Advanced Concepts and Paradigms*, AIAA, Washington, DC.

Levis, A. H. (1995). Human Interaction with Decision Aids: A Mathematical Approach. In W. B. Rouse (Ed.) *Human/Technology Interaction in Complex Systems*, JAI Press, Greenwich, CT.

Lu, Z. (1992). Coordination in Distributed Intelligence Systems. Report GMU/C3I-120-TH, M.S. Thesis, Center of Excellence in Command, Control, Communications, and Intelligence, George Mason University, Fairfax, VA.

Marschak, J. and R. Radner (1972). *The Economic Theory of Teams*, Yale University Press, New Haven, CT.

Martin, P. J. and A. H. Levis (1987). Measures of Effectiveness and C3 Testbed Experiments. *Proc. 1987 Symposium on Command and Control Research*, National Defense University, Fort Mc Nair, Washington, DC.

Mesarovic, M. D., D. Macko, and Y. Takahara (1970). *Theory of Hierarchical, Multilevel Systems*, Academic Press, NY.

Minsky, M. (1986).*The Society of Mind*, Simon and Schuster, NY.

Monguillet, J. M., and A. H. Levis (1993). Modeling and Evaluation of Variable Structure Organizations. In C. R. Jones (Ed.) *Command Control and Communications: Advanced Concepts and Paradigms*, AIAA, Washington, DC.

Monguillet, J.-M. (1987). Modeling and Evaluation of Variable Structure Organizations. Report LIDS-TH-1730, Laboratory for Information and Decision Systems, MIT, Cambridge, MA.

Neapolitan, R.E. (1990*). Probabilistic Reasoning in Expert Systems: Theory and Algorithms,* Wiley, NY.

Orasanu, J. and E. Salas (1993). Team Decision Making in Complex Environments. In G. A. Klein, J. Orasanu, R. Calderwood, and C. E. Zsambok (Eds.) *Decision Making in Action: Models and Methods.* Ablex Publishing Corp., Norwood, NJ.

Pearl, J. (1988). *Probabilistic Reasoning In Intelligent Systems: Networks of Plausible Inference*, Morgan Kaufmann, San Mateo, CA.

Pete, A., K.R. Pattipati, and D.L. Kleinman, (1995). Designing Organizations with Congruent Structures. *Proc. of the First International Symposium on Command and Control Research and Technology*, National Defense University, Ft. McNair, Washington D.C. pp. 197-209.

Pete, A., K.R. Pattipati, and, D.L. Kleinman (forthcoming). Optimization of Decision Networks in Structured Task Environments. *IEEE Trans. Syst., Man , Cybern.*

Peterson, J. L. (1981). *Petri Net Theory and the Modeling of Systems*, Prentice Hall, Englewoods Cliffs, NJ.

Reisig, W. (1985). *Petri Nets, an Introduction.* Springer-Verlag, Berlin, Germany.

Remy, P. (1986). On the Generation of Organizational Architectures Using Petri Nets. Report LIDS-TH-1630, M.S. Thesis, Laboratory for Information and Decision Systems, MIT, Cambridge, MA.

Remy, P. and A. H. Levis A.H. (1988). On the Generation of Organizational Architectures Using Petri Nets. In G. Rozenberg (Ed.) *Advances in Petri Nets 1988*, Lecture Notes in Computer Science, Springer Verlag, Berlin, Germany.

Salas, E., T. L. Dickinson, S. A. Converse, and S. I. Tannenbaum (1992). Toward an Understanding of Team Performance and Training. In R. W. Swezey and E. Salas (Eds.) *Teams: Their Training and Performance.* Ablex Publishing Corp., Norwood, NJ

Shachter, R. D. (1986). Evaluating Influence Diagrams. *Operations Research*, Vol. 34, pp. 871-882.

Shachter, R. D. (1990). An Ordered Examination of Influence Diagrams. *Networks*, Vol. 20, pp. 535-563.

Tatman, J.A., and R.D. Shachter, (1990.) Dynamic Programming and Influence Diagrams. *IEEE Trans. Syst., Man, Cybern.* Vol. SMC-20, No. 2, pp. 365-379.

Vroom, V. E., and P. W. Yetton (1973). *Leadership and Decision Making*, University of Pittsburgh Press, Pittsburgh, PA.

Zaidi, A.K. (1991). On the Generation of Multilevel Distributed Intelligent Systems Using Petri Nets. Report GMU/C3I-112-TH, M.S. Thesis, Center of Excellence in Command, Control, Communications, and Intelligence, George Mason University, Fairfax, VA.

Zaidi, A.K (1994). Validation and Verification of Decision Making Rules. Report GMU/C3I-155-TH, Ph. D. Dissertation, Center of Excellence in Command, Control, Communications, and Intelligence, George Mason University, Fairfax, VA.

Zaidi, A. K., and A. H. Levis (1995). Algorithmic Design of Distributed Intelligence System Architectures. In M. M. Gupta and N. K. Sinha (Eds.), *Intelligent Control Systems: Theory and Practice*, IEEE Press, NY.

5. MEETINGS

Dr. Levis visited the Naval Postgraduate School, Monterey, CA in August 1995 and briefed the staff on GMU research

6. CHANGES

Changes in the scope of work of this project, as a result of two contract modifications, have been documented in section 2.

7. CURRENT RESEARCH PERSONNEL

Prof. Alexander H. Levis	Principal Investigator
Prof. Dennis Buede	
Dr. Abbas K. Zaidi	Consultant
Mr. Lee Wagenhals	Research Instructor (Ph.D.)
Mr. Didier Perdu	Research Instructor (Ph.D.)
Mr. Eric Tzibertsopoulos	Graduate Research Asst. (MS)
Mr. Bruce Bernier	Undergraduate Research Assistant
Ms. Audrey Molsky	Undergraduate Research Assistant
Ms. Etsiwohot Dinka	Undergraduate Research Assistant

8. DOCUMENTATION

1. Hedy L. Rashba (1993). Problems in Concurrency and Coordination in Decision Making Organizations. Report GMU/C3I-143-R, C3I Center, George Mason University, Fairfax, VA.

2. A. H. Levis (1993). Adaptive Decision Making and Coordination in Variable Structure Organizations. Report GMU/C3I-147-IR, C3I Center, George Mason University, Fairfax, VA.

3. Zhenyi Jin (1994). Deadlock and Trap Analysis in Petri Nets. MS Thesis, Systems Engineering Department, George Mason University, Fairfax, VA.

4. A. H. Levis and D. M. Perdu (1994). Object Oriented Design of Decision Making Organizations, *Proc. 1994 The First Workshop on Command Informations Systems,* Oxon, UK. pp. 372-384. Defence Research Agency. Malvern

5. A. K. Zaidi (1994). Validation and Verification of Decision Making Rules, Report GMU/C3I-155-TH, C3I Center, George Mason University, Fairfax, VA.

6. A. H. Levis and D. M. Perdu (1994). Object Oriented Design of Decision Making Organizations, A. H. Levis & I. S. Levis (Editors), *The Science of Command and Control: Part III, Coping with Change,* AFCEA International Press, Fairfax, VA.

7. A. K. Zaidi and A. H. Levis (1995). Rule Decomposition and Validation for Distributed Decision Making, *Proc. 1995 First International Symposium on Command and Control Research and Technology,* National Defense University, Washington, DC. pp. 210-217.

8. A. K. Zaidi and A. H. Levis (1995). Validation and Verification of Decision Making Rules, *Proc. 1995 6th IFAC/IFIP/IFORS/IEA Symposium on Analysis, Design and Evaluation of Man-Machine Systems,* MIT, Cambridge, MA. pp. 53-60.

9. A. H. Levis, D. M. Perdu and A. K. Zaidi (1994). Adaptive Decision Making and Coordination in Variable Structure Organizations. Report GMU/C3I-151-IR, C3I Center, George Mason University, Fairfax, VA.

10. A. H. Levis, D. M. Perdu & A. K. Zaidi (1995). CAESAR II: Computer Aided Evaluation Of System Architectures, Report GMU/C3I-162-R, C3I Center, George Mason University, Fairfax, VA.

11. A. K. Zaidi and A. H. Levis (1995). Object Oriented Design of a Multilevel Hierarchical Organization Structure, *Proc. US/Portugal Workshop On Underwater Vehicles and Intelligent Control,* Lisbon, Portugal. University of So. Louisianna Press.

12. A. K. Zaidi and A. H. Levis (1995).Validation and Verification of Decision Making Rules. Accepted for publication in *Automatica.*

13. Zaidi, A. K. & A. H. Levis (1996). On Generating DIS Architectures Using Genetic Algorithms, Paper GMU/C3I-166-P, C3I Center, George Mason University, Fairfax, VA.